Web开发实用技术基础

刘於勋◎主　编
韩　璐　胡江汇　苏小玲◎副主编





清华大学出版社

Web 开发实用技术基础

刘於勋　主编

韩　璐　胡江汇　苏小玲　副主编

清华大学出版社

北　京

内 容 简 介

本书全面、详细地介绍了 Web 开发实用技术基础知识，内容包括 Web 的相关概念（Web 服务、Web 数据库、ASP.NET 技术、JSP 技术、Ajax 技术、语义 Web 服务技术、Web3D 技术）、HTML 及 XML 基础、网页与网站设计基础、Dreamweaver CS4 工具、Web 客户端脚本语言设计（JavaScript 语言基础、浏览器对象模型 BOM、文档对象模型 DOM、Ajax 技术）、基于 ASP.NET 的服务器端程序设计以及基于 JSP 的服务器端程序设计等。

全书采用基本概念与实际应用相结合的方式组织内容，使读者能够在深刻理解和全面领会 Web 开发程序设计特点和风格的基础上，切实掌握面向对象的 Web 开发技术。

本书通俗易懂、结构合理、叙述简洁，每章都有小结和思考题，可帮助读者尽快掌握所学知识。本书可作为高等院校计算机或其他相关专业开设 Web 开发技术课程的教材，也可作为自学 Web 开发技术的参考书。

图书在版编目（CIP）数据

Web 开发实用技术基础/刘於勋主编. —北京：清华大学出版社，2010.12

ISBN 978-7-302-23930-7

I. ①W…　II. ①刘…　III. ①主页制作–程序设计　IV. ①TP393.092

中国版本图书馆 CIP 数据核字（2010）第 195437 号

责任编辑：郭新义　朱　俊
封面设计：张　岩
版式设计：牛瑞瑞
责任校对：王　云
责任印制：
出版发行：清华大学出版社
http://www.tup.com.cn
社 总 机：010-62770175
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
地　　址：北京清华大学学研大厦 A 座
邮　　编：100084
邮　　购：010-62786544
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185×260　**印　　张：**22.5　**字　　数：**516 千字
版　　次：2010 年 12 月第 1 版　**印　　次：**2010 年 12 月第 1 次印刷
印　　数：1～4000
定　　价：38.00 元

产品编号：037333-01

前　　言

由于 Internet 采用超文本和超媒体的组合方式将信息的链接扩展到整个 Internet 上，使得人们足不出户通过浏览器就能够访问到 Internet 上丰富多彩的网络资源。而 Web 就是一种超文本信息系统，它使文本不再固定在某一个位置，而是可以从一个位置跳到另外的位置。本书介绍了 Web 实用技术，其内容主要包括 Web 技术概述、Web 页面的基础知识（HTML 和 XML）、网页制作工具（Dreamweaver CS4）的操作方法、JavaScript 脚本在 Web 页面中的应用、运用 APS.NET 技术、JSP 技术在服务端开发应用程序等。

本书适合学过面向对象程序设计的读者使用。因为无论是客户端程序设计（JavaScript 脚本语言），还是服务器端程序设计（ASP、JSP），都是采用面向对象程序设计方法进行设计的，其目的是使读者学会使用类编程，领会面向对象程序设计的优势，进而牢固掌握面向对象程序设计的各种方法。

本教材的特点如下：

1．组织结构新颖

本书首先介绍 Web 的相关概念、Web 服务、Web 数据库、ASP.NET 技术、JSP 技术、Ajax 技术、语义 Web 服务技术、Web3D 技术等，使读者首先对 Web 开发技术有比较系统、全面的了解和认识，并对 Web 技术产生兴趣。

2．突出 Java 面向对象设计

全书紧紧围绕与 Java 面向对象相关的理论知识由浅入深地展开，由 JavaScript 脚本语言、Java 基础介绍，过渡到 JSP 编程，突显 JSP 服务器端程序的设计方法，重点介绍 JSP 的概况、JDK 的获取和安装、Tomcat 服务器的下载和配置、JSP 开发环境 Eclipse 的配置、JSP 的基础知识、JSP 的内置对象、4 种 JDBC 驱动程序（分别为 JDBC-ODBC Bridge、JDBC-NativeAPI Bridge、JDBC-middleware 和 PureJDBC-Driver）、DriverManager 类方法、使用 JDBC 连接 4 种数据库的方法、用 JavaBean 封装服务器数据库等内容，最后用一个会员管理系统作为范例，详细介绍如何在 Eclipse 环境下编写 JSP 动态网页。

3．示例通俗易懂

书中所举示例实用、易理解、易掌握，均已通过上机调试、运行。

本书由河南工业大学信息科学与工程学院刘於勋主编，韩璐、胡江汇、苏小玲为副主编。参与编写的还有高山和孙宜贵等。其中第 1、5 章由刘於勋编写，第 2 章由苏小玲编写，第 3 章由韩璐编写，第 4 章由高山编写，第 6 章由孙宜贵编写，第 7 章由胡江汇编写。全

书由刘於勋统稿。

本书在编写的过程中得到了兄弟院校和同行们的支持和关心，他们对本书的编写提出了许多宝贵的建议和意见，在此对审定本教材编写大纲的专家们及在全书编写过程中给予帮助的同事们表示感谢。

由于编者水平有限，加之时间仓促，书中难免有不妥之处，敬请广大专家、读者批评指正。

编　者

目　　录







第 1 章　Web 概述

当前，Internet 已渗入到人们生活中的各个领域，如网上学习、办公、购物、聊天、玩游戏、看电视、娱乐、发布信息、检索信息等，它正改变着人们的工作、生活及学习方式。因此，了解 Internet 所提供的 Web 服务和技术对于使用者来说是非常有必要的。本章将重点介绍 Web 概念、Web 工作原理、Web 服务、Web 技术及其发展趋势。

1.1　Web 概念

Web 是 World Wide Web（环球信息网）的简称，中文意思是“万维网”。它起源于 1989 年 3 月，由欧洲粒子物理实验室 CERN（the European Laboratory for Particle Physics）所发展出来的主从结构分布式超媒体系统。人们通过访问 Web 服务器，就可以很迅速、方便地访问到所需要的信息资料。由于用户在使用 Web 浏览器访问信息资源的过程中，无须关心技术性的细节，就可以得到想要的信息，所以，Web 一经推出，就备受人们的关注，并得到了迅速、蓬勃的发展。Web 是一种体系结构，通过它可以访问分布于 Internet 主机上的链接文档，它有多层含义：Web 是 Internet 提供的一种服务；是存储在 Internet 网络中数量巨大的文档的集合；是由彼此关联的文档组成的海量信息源，这些文档称为主页或页面；Web 的内容保存在 Web 站点中，用户可以通过浏览器访问 Web 站点。那么 Web 的定义是什么呢？

1.1.1　Web 定义

Web 是英国人 Tim Berners-Lee 于 1989 年在欧洲共同体的一个大型科研机构任职时发明的。互联网上的资源可以在一个网页中比较直观地表示出来，网络资源之间在网页上可以相互链接。Web 就是一种超文本信息系统，它使得文本可以从一个位置跳到另外的位置。如果想了解某一个主题的内容，只要在这个主题上单击一下，就可以跳转到包含这一主题的文档中。

1.1.2　Web 的五要素

构成 Web 的 5 个要素分别为 URL、HTTP、HTML、Web 浏览器和 Web 服务器。下面分别介绍这 5 个要素的具体含义。

1. URL

URL（Universal Resource Locator）是统一资源定位器。它是方案集，包含如何访问 Internet 上的资源的明确指令。另外，URL 是统一的，无论寻找哪种特定类型的资源（如音乐、邮件、文件、新闻等）或描述通过哪种机制获取网络资源，都采用相似的 URL 格式。

URL 格式为：

protocol:// hostname[:port] / path / [;parameters][?query]

其中，方括号（[]）中的为可选项，其中的参数解释如下。

- protocol://：通信协议方案。常用协议有如下几种形式。
  - ☑ file://：指本地计算机上的文件。
  - ☑ ftp://：指通过 FTP 协议访问 FTP 服务器上的资源。
  - ☑ http://：指通过 HTTP 协议访问网站资源。
  - ☑ https://：指通过 HTTPS 安全协议访问网站资源。
  - ☑ mms://：指通过 MMS 协议访问 Windows Media 发布点上的单播内容。
  - ☑ ed2k://：指通过支持 ED2K 协议的 P2P 软件访问网络资源。
- hostname：DNS（域名系统）中的主机名及其 IP 地址。
- :port：端口号（整数，可选），省略时使用协议的默认端口，如 ftp 协议的默认端口为 21。
- path：路径，由零或多个“/”符号隔开的字符串，一般用来表示主机上的一个目录或文件位置。
- ;parameters：用于指定特殊参数（可选）。
- ?query：查询（可选），用于给动态网页（如使用 CGI、ISAPI、PHP、JSP、ASP、ASP.NET 等技术制作的网页）传递参数，如果有多个参数，用“&”符号隔开，每个参数的名和值用“=”符号隔开。

2. HTTP

HTTP（Hypertext Transfer Protocol）是超文本传输协议，是用于从 Web 服务器传输超文本到本地浏览器的传送协议。它不仅保证计算机正确、快速地传输超文本文档，还确定传输文档中的哪一部分，以及哪一部分内容首先显示（如文本先于图形）等。

3. HTML

HTML（Hypertext Markup Language）是超文本标记语言，它之所以称为超文本标记语言，是因为文本中包含了“超链接”点。所谓超链接，就是一种 URL 指针，通过单击它，浏览器可以方便地获取新的网页。HTML 是一种标准，它通过标记符号来标记要显示的网页中的各个部分。网页文件本身是一种文本文件，通过在文本文件中添加标记符，可以告诉浏览器如何显示其中的内容（如文字如何处理、画面如何安排、图片如何显示等）。浏览器按顺序阅读网页文件，然后根据标记符解释和显示其标记的内容，对书写出错的标记不指出其错误，且不停止其解释执行过程，编制者只能通过显示效果来分析出错原因和出错部位。



但需要注意的是，对于不同的浏览器，对同一标记符可能会有不完全相同的解释，因而可能会有不同的显示效果，由此可见，网页的本质就是 HTML，通过结合使用其他的 Web 技术（如脚本语言、CGI、ASP、JSP 等），可以创造出功能强大的动态网页。因而，HTML 是 Web 编程的基础，也可以说万维网是建立在超文本基础之上的信息网络。

4．Web 浏览器

Web 浏览器（Web Browser）实际上是一个软件程序，用于与 Web 服务器建立连接，并与之进行通信。它可以通过访问 DNS 系统中的主机名确定信息资源的位置，并将用户感兴趣的信息资源返回到浏览器中的网页里，对 HTML 文件进行解释，然后将文字图像或者多媒体信息还原出来。目前，国内外常用的浏览器为 IE（Internet Explorer 微软）、NN（Netscape Navigator 网景）、Mosaic（美国伊利诺州的伊利诺大学的 NCSA 组织）、Opera（Opera Software ASA 公司）、Chrome（谷歌）、Safari（苹果）、Firefox（火狐浏览器）、TT（腾讯）、Maxthon（遨游）和 360 安全等。

5．Web 服务器

Web 服务器（Web Server）指驻留在 Internet 上的某种类型的计算机程序，它可以解析 HTTP 协议。当 Web 服务器接收到一个 HTTP 请求（Request）时，会返回一个 HTTP 响应（Response）。为了处理一个请求（Request），Web 服务器可以响应（Response）一个静态页面或图片，进行页面跳转（Redirect），或者把动态响应（Dynamic Response）的产生委托（Delegate）给一些其他的程序，如 CGI 脚本、JSP（Java Server Pages）脚本、Servlet、ASP（Active Server Pages）脚本，无论这些脚本的目的如何，服务器端的程序通常产生一个 HTML 的响应（Response）来让浏览器可以浏览。Tomcat、NCSAhttpd 和 Apache 服务器是基于 UNIX 和 Linux 平台的，而 IIS 是基于 Windows 平台的浏览器。下面简要介绍这几种 Web 服务器。

（1）Microsoft IIS

IIS（Internet Information Server）是 Microsoft 的 Web 应用程序服务器产品，它允许在公共 Intranet 或 Internet 上发布信息。IIS 是目前最流行的 Web 服务器产品之一，很多著名的网站都建立在 IIS 的平台上。IIS 提供了 Internet 服务管理器，它是基于图形界面的管理工具，主要可用于监视配置和控制 Internet 服务。

IIS 是一种 Web 服务组件，提供 Web 服务器、FTP 服务器、NNTP 服务器和 SMTP 服务器等多种功能，主要用于浏览网页、传输文件、新闻服务和发送邮件。通过 IIS 服务器可以很容易地在 Internet 或 Intranet 上发布信息。

（2）IBM WebSphere

IBM WebSphere 是一种功能完善、开放的 Web 应用程序服务器，它基于 Java 的应用环境，用于建立、部署和管理 Internet 和 Intranet Web 应用程序。WebSphere 服务器针对以 Web 为中心的开发人员，支持 HTTP 协议和动态网页编程技术。IBM 为用户从简单的 Web 应用程序转移到电子商务世界提供 WebSphere 产品系列支持。

（3）BEA WebLogic

BEA WebLogic 是一种多功能、基于 Java 应用环境的 Web 应用服务器，为企业构建自己的应用提供了坚实的基础。各种应用开发、部署所有关键性的任务、集成各种系统和数据库、提交服务、跨 Internet 协作，起始点都是 BEA WebLogic 服务器。BEA WebLogic 服务器在使应用服务器成为企业应用架构的基础方面继续处于领先地位，为构建集成化的企业级应用提供了稳固的基础。由于它具有全面的功能，所以，基于 Internet 的企业都选择它来开发、部署最佳的应用。

（4）Apache

Apache 是一款使用最广的、基于 UNIX、Windows、Linux 系统平台的 Web 应用服务器。它源于 NCSAhttpd 服务器，当 NCSA Web 服务器项目停止后，那些使用 NCSA Web 服务器的人们开始交换用此服务器的补丁，这也是 Apache 名称的由来（pache，补丁）。世界上很多著名的网站都是 Apache 的产物，它的成功主要在于它的源代码开放、有一支开放的开发队伍、支持跨平台的应用及可移植性等。

（5）Tomcat

Tomcat 是一个开放源代码、运行 Servlet 和 JSP Web 应用软件的基于 Java 的 Web 应用软件容器。它是基于 Apache 许可证下开发的自由软件，是 Java Servlet 2.2 和 JavaServer Pages 1.1 技术的标准实现，是完全重写的 Servlet API 2.2 和 JSP 1.1 兼容的 Servlet/JSP 容器。Tomcat 使用了 JServlet 的一些代码，特别是 Apache 服务适配器。随着 Catalina Servlet 引擎的出现和 Tomcat 6.0 性能的提升，使它成为一个值得考虑的 Servlet/JSP 容器，因此，目前许多 Web 服务器都采用 Tomcat。

Web 服务器有 3 个主要发展趋势：

（1）从 HTML 到 XML（Extensible Markup Language，可扩展标记语言）

HTML 作为 Web 的开发语言，被称为“第一代 Web 语言”，它对 Web 应用的发展起到了关键性的作用。但它的致命缺点是只适合于人与计算机的交流，而不适合计算机与计算机的交流。HTML 通过大量的标记来定义文档内容的表现方式，它仅描述了应如何在 Web 浏览器页面上布置文字、图形，而没有对 Internet 中最重要的信息含义本身进行描述。Web 服务器向 Web 浏览器提供的信息都来自有一定结构的数据库，在数据库中，为了检索和管理的方便，信息按照它本身的意义（如学号、姓名、班级、成绩等）被存放在相应的字段里，一旦这些数据被调出来，经过 CGI、ASP.NET、JSP、PHP 等转换成 HTML 后，其原来的意义无法转移到 HTML 标记中来，用户也就无法按照信息本来的意义去阅读。此外，HTML 还有一个问题，就是它的标记集合是固定的，所以用户不能根据自己的需要增加标记。

由于操作系统以及数据库的不同，不同的系统及应用层面之间要想互相理解对方的数据格式是相当困难的，这就需要一种新技术或标准，能够将最初保存在数据库服务器中的原始数据结构在不同的系统层面共享，这种新技术就是 XML。

W3C 对 XML 作了这样的描述——XML 描述了一类被称为 XML 文档的数据对象，并部分描述了处理它们的计算机程序的行为。XML 是 SGML 的一个应用实例。从结构上说，XML 文档遵从 SGML 文档标准。同 HTML 一样，XML 也是一种基于文本的标记语言，都是从 SGML（Standard Generalize Markup Language，标准通用标记语言）发展而来的，二者



的区别在于：XML 可以根据要表现的文档，自由地定义标记，来表现具有实际意义的文档内容，例如，可以定义〈文档名称〉〈/文档名称〉这样具有实际意义的标记。而且，XML 不像 HTML 那样具有固定的标记集合，它实际上是一种定义语言的语言，也就是说使用 XML 的用户可以定义无穷的标记来描述文档中的任何数据元素，将文档的内容组织成丰富的、完整的信息体系。总而言之，XML 具有便于存储的数据格式、可扩展性、高度结构化以及方便的网络传输四大特点，这些特点为创建开放、高效、可扩展、个性化的 Web 应用提供了坚实的基础。

（2）从有线到无线

电子商务正在从台式机向着更为广泛的无线设备发展，Cahners In-Stat 市场分析家预测，世界范围内的无线用户的人数将会从 2000 年的 2.71 亿增加到 2004 年的 13 亿。Aberdeen 集团的研究主任 Darcy Fowkes 认为，采用无线方式进行电子商务的优势并不仅仅在于方便，它还可以节约公司的财力，而且移动办公能使工作更加高效。

然而，由于多种无线网络类型、标记语言、协议和无线设备并存的复杂情况，使得网络内容和数据转换成能够被无线设备所识别的格式并不容易。目前，许多企业都在致力于开发能够把应用程序以及互联网内容扩展到无线设备上的产品。

例如，IBM 新版本的 WebSphere Transcoding Publisher 3.5 增加和改进了许多新的特性，可以将企业内部网上的数据翻译到多种无线设备上。该版本中新的特性包括对更多的无线设备、数据格式的支持，以及语言翻译功能。它基于 Java 架构，能把用 HTML 和 XML 等标记语言编写的应用程序和数据转换成 WML、HDML（Handheld Device Markup Language）和 IMode 等无线设备所能识别的格式，这样，通过手持设备就可以访问互联网上的信息。

无线设备厂商 Mobilize 也推出了 Mobilize Commerce 产品，帮助企业进入无线网络。该软件可以通过无线连接的方式访问企业的内部系统，远程地实现订单发送，并进行确认。Mobilize Commerce 可以充分利用 XML 对信息进行格式转换，以适合于无线设备，这些无线设备包括笔记本电脑、个人数字助理、无线电话、网络电话和双向寻呼等。

（3）从无声到有声

世界上现在有几十亿个电话终端，有超过 5 亿的移动电话，而就人自身的交流习惯来看，人们也更愿意利用听和说的方式进行交流。

目前，文本语音转换器（Text to Speech，TTS）的研究工作已经取得了很大的进步，实现了自动的语言分析理解，并允许 TTS 的使用者增加更多的韵律、音调在讲话中，使 TTS 系统的发声更接近人声。在自动语音识别系统（ASR）领域中，自动语音识别系统正从整个词的模仿匹配，向音素层次的识别系统方向发展。自动语音识别系统的词汇表由一个基于声音片断的字母表构成，而且这种词汇表是受不同语言限制的。基于这种方式，在一个宽广的声音行列里，讲话能被识别系统发现和挑拣出来，并加以识别。并且在识别一个词的时候，每一个音素将从系统的输入中挑拣出来，拼接组合后与已经有的音素和词语模板进行比较，来产生需要的模板。音素的识别大大减轻了 ASR 对讲话者的依赖性，并且使得它可以非常容易地建立大型的和容易修改的语音识别字典，从而满足不同应用市场的需求。

Web 语音发展的另一方面是 Voice XML（Voice Extensible Markup Language，语音可扩展标记语言）的进展。Voice XML 的主要目标是要将 Web 上已有的大量应用、丰富的内容，

让交互式语音界面也能够全部享受。Web 服务器处理一个来自客户端应用的请求，这一请求经过了 Voice XML 解释程序和 Voice XML 解释程序语境处理，作为响应，服务器产生出 Voice XML 文件，在回复中，要经过 Voice XML 解释程序的处理。Voice XML 1.0 规范基于 XML，为语音和电话应用的开发者、服务提供商和设备制造商提供了一个智能化的 API。Voice XML 的标准化将简化 Web 上具有语音响应服务的个性化界面的创建，使人们能够通过语音和电话访问网站上的信息和服务，像使用 CGI 脚本一样检索中心数据库、访问企业内部网、制造新的语音访问设备。Voice XML 的执行平台上面加载了相应的软件和硬件，如 ASR、TTS，从而实现语音的识别以及文本和语音之间的转化。2000 年 5 月 23 日，W3C 接受了语音可扩展标记语言 Voice XML 1.0 作为实例。IBM、NOKIA、Lucent、Motorola 等著名厂商都已经开发出相应的支持 Voice XML 的产品。

1.1.3 Web 特性

Web 具有 5 个特性，下面分别进行介绍。

（1）导航性

导致 Web 非常流行的一个很重要的原因是其非常易于导航，只需要从一个超链接跳到另一个超链接，就可以访问各站点所提供的服务，浏览所需要的多媒体信息。同时，在同一页面上可以显示色彩丰富的图形和文本。

（2）与平台无关性

无论是在 Windows 平台上，还是在 UNIX、Linux、Macintosh 平台上，都可以通过浏览器（Browser）的软件访问 Web 资源。

（3）分布性

大量的视频、音频、文本信息存放在不同的站点上，只要在浏览器中单击这个站点，就可以将不同地理位置上的网站信息组合成逻辑上统一的资源，供人们使用。

（4）动态性

由于 Web 站点上的信息经常发生变动，需要及时更新，所以是动态的、可变的。Web 动态的特性还表现在人与计算机可以进行交互。

（5）交互性

交互性首先表现在超链接上，浏览顺序和所指的站点完全由人来决定；其次是用户通过填写 FORM 可以向服务器提交请求，服务器可以根据用户的请求返回相应信息。

1.2 Web 工作原理

Web 是基于浏览器/服务器的一种体系结构，通过浏览器向服务器发送请求，要求执行某项任务，而服务器执行此项任务，并向浏览器返回响应。Web 工作原理如图 1-1 所示。



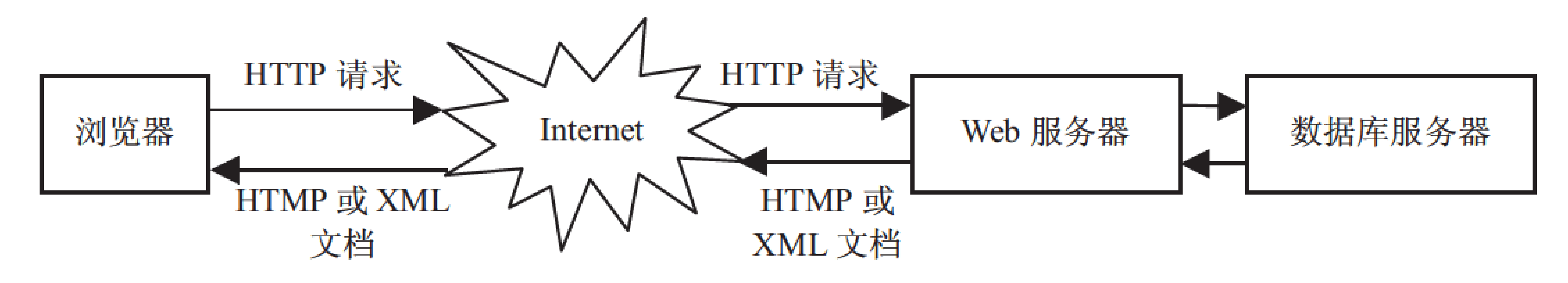


图 1-1　Web 工作原理

在浏览器/服务器模式体系结构系统中，用户通过浏览器向分布在网络上的许多服务器发出请求，服务器对浏览器的请求进行处理，将用户所需信息返回到浏览器。而如数据请求、加工、结果返回、动态网页生成、对数据库的访问和应用程序的执行等工作全部由 Web 服务器完成。

Web 服务器向浏览器提供服务的过程如下：

（1）用户启动浏览器程序，在浏览器中指定一个 URL，浏览器便向该 URL 所指向的 Web 服务器发出请求。

（2）Web 服务器接到浏览器请求后，把 URL 转换成页面所在服务器上的文件路径名。

（3）若 URL 指向的是普通的 HTML 文档，Web 服务器直接将它送给浏览器。

（4）如果 HTML 文档中嵌有 ASP 和 JSP 程序，Web 服务器就运行 ASP 或 JSP 程序，并将结果传送至浏览器。

（5）URL 也可以指向 VRML（Virtual Reality Modeling Language，虚拟现实建模语言）文档。

1.3　Web 服务

1.3.1　什么是 Web 服务

到底什么是 Web 服务？不同的组织和部门对 Web 服务的定义略有差异。

1. W3C（国际标准化组织）定义

Web 服务是一个通过 URL 识别的软件应用程序，其界面及绑定能用 XML 文档来定义、描述和发现，使用基于 Internet 协议上的消息传递方式与其他应用程序进行直接交互。

2. Microsoft 定义

Web 服务是为其他应用提供数据和服务的应用逻辑单元，应用程序通过标准的 Web 协议和数据格式获得 Web 服务，如 HTTP、XML 和 SOAP 等，每个 Web 服务的实现是完全独立的。



3．IBM 认为

Web 服务是一种自包含、自解释、模块化的应用程序，能够被发布、定位并且从 Web 上的任何位置进行调用。Web 服务可以执行从简单的请求到错综复杂的商业处理过程中的任何功能。理论上来讲，一旦对 Web 服务进行了部署，其他 Web 服务应用程序就可以发现并调用已部署的服务。

4．UDDI 规范中定义

Web 服务是指由企业发布的完成其特别商务需求的在线应用服务，其他公司或应用软件能够通过 Internet 来访问并使用这项应用服务。

由此可以看出，这些定义各有侧重，但有几点是一致的。首先，它是由企业驱动和应用驱动而产生的；其次，它具有分布性、松散耦合、可复用性、开放性以及可交互性等特性。

Web 服务技术促进了 Internet 上企业之间的协作，使用 Web 服务可以使合作伙伴的信息系统之间更容易地进行通信。如何集成来自不同企业的服务、又便于使用 Web 服务？众多组织纷纷致力于 Web 服务技术的研究，进行标准协议的制定，提供 Web 服务的创建工具和解决方案。

Web 服务不同于已有的构件对象模型以及相关的对象模型协议，如 CORBA 和 IIOP（Internet Inter-ORB Protocol）、COM 和 DCOM 以及 Java 和 RMI（Remote Method Invocation）。Web 服务可以用任何语言编写，并且可以使用 HTTP 或 XML 访问。从技术上看，一个 Web 服务是一个由内容、应用代码、过程逻辑或者这些部分的任意组合所构成的 XML 对象，并且可以通过任何 TCP/IP 网络访问，只要网络中使用 SOAP 标准集成，使用 WSDL 标准进行自描述，使用 UDDI 标准在一个公共的或者私有的目录中注册和发现。Web 服务由多层构成（如图 1-2 所示），这些层堆叠在一起形成了发现和调用一个独立的 Web 服务所提供功能的标准机制的基础。即 Web 服务栈以层次结构来表示，高层在低层的基础之上构建。

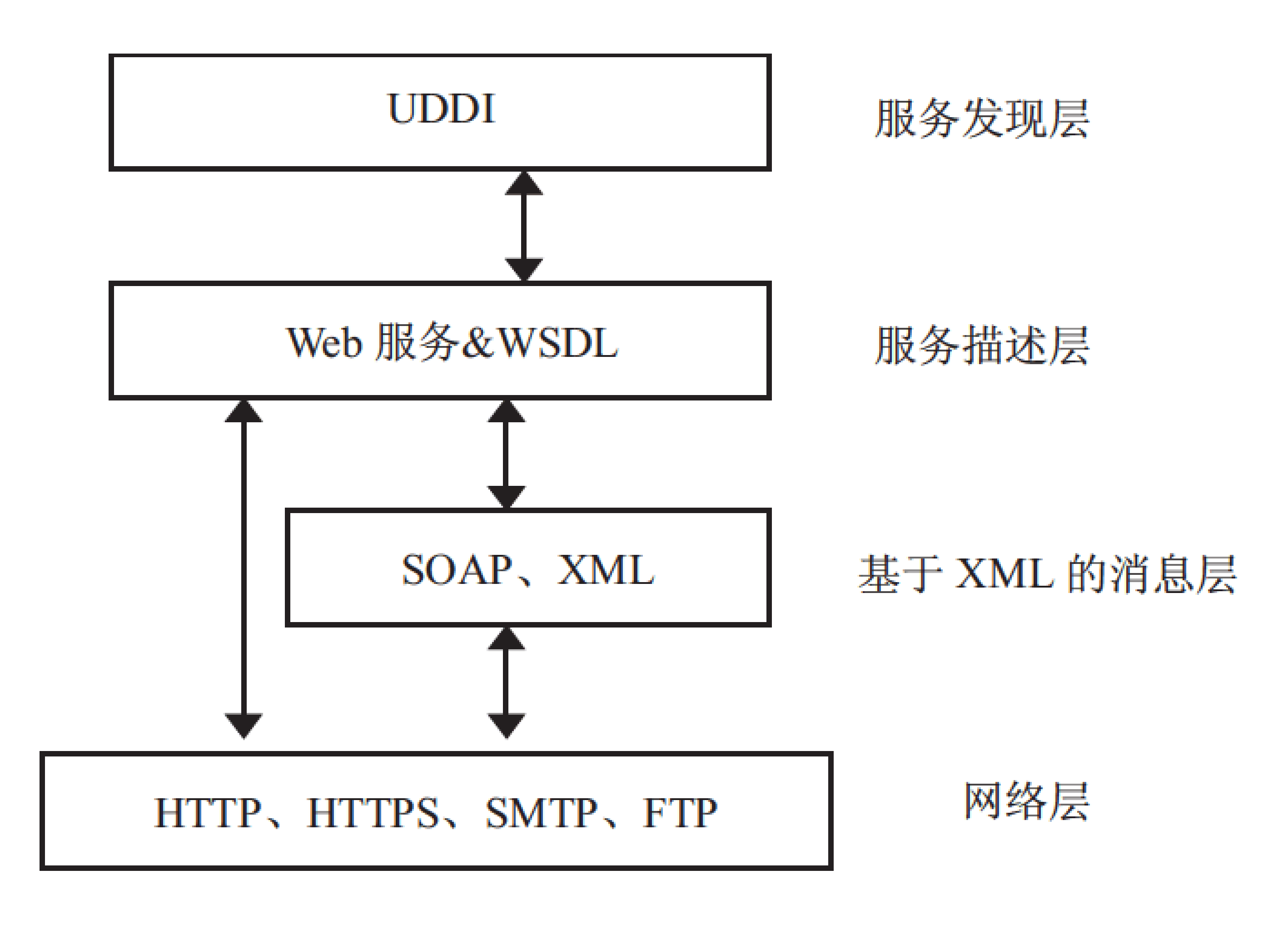


图 1-2　Web 服务的多层构成



图 1-2 中网络层中的 HTTP、HTTPS、SMTP、FTP 协议提供了分布式应用之间的通信机制；消息层中的 XML 定义了数据交换和描述的格式，SOAP 是调用 Web 服务的协议；服务描述层中的 WSDL 描述 Web 服务的格式；而服务发现层中的 UDDI 则是注册、查找和使用 Web 服务的中枢组织。由 XML、SOAP、WSDL、UDDI 构成了 Web 服务的关键技术，下面分别介绍其具体含义。

（1）XML（Extensible Markup Language，可扩展标记语言）

该层包括数据表示、数据格式和消息传输协议。XML 为信息交换定义了描述和格式。由于 HTTP 是一种基于文本的协议，因此缺乏表示 RPC 消息中的参数值的机制，被 XML 取代作为 Web 服务的一个重要的数据描述语言。XML 是一种元语言，可以通过标准的编码和格式化信息的方法进行跨平台的数据交换。XML 允许数据被串行化为易于被任何平台解码的消息格式，提供了在网络应用之间交换结构化数据的机制。

XML 采用纯文本表示，设计的初衷是为了存储、传送和交换数据。XML 是一种标记语言，标记在 XML 中不是预先确定的，而必须由使用者自己定义。XML 允许使用者自由发表有用的信息，不仅可以是有关数据结构的，也可以是关于数据意义的。另外，XML 文档的结构、内容和外观可以作为 3 个不同的部分进行维护，提供了更高的独立性。

对于数据表示层来说，可扩展性是一个关键因素。为了支持可扩展性，Web 服务需要一种机制以避免名字冲突，并允许一个程序只处理自己所关心的元素。XML 名空间（namespaces）提供了一种简单、通用的方式以区分相同名字的元素或属性。为了支持可扩展性，XML 中的每个元素和属性都有一个相关的名空间 URL。Web 服务需要一种方法定义 Web 服务消息中使用的数据类型。XML Schema 规范标准化了一个描述 XML 数据类型的符号集，还定义了一个内置简单数据类型的集合和在各 XML 文档中建立元素类型的机制。XML Schema 规定了 XML 的文档的逻辑结构，定义了元素、元素属性以及元素和元素属性之间的关系。

XML 还在不断地发展。需要说明的是，XML 本身只是一种标记语言，只是进行描述，并不提供商务逻辑，Web 服务提供对这些逻辑的访问。这也是为什么 Web 服务的更高层的、基于 XML 的概念同样非常重要的原因。

（2）SOAP（Simple Object Access Protocol，简单对象访问协议）

SOAP 是目前被广泛接受的消息传输协议，是一个为信息交换设计的轻量协议，用于在网络应用程序之间交换结构化数据，是一种基于 XML 的机制。SOAP 主要是在分布的、分散的环境中提供了一个跨 Internet 调用服务的框架结构，并提供了独立于编程语言和分布对象底层基础结构的跨平台集成机制。SOAP 代表了 XML-RPC 的发展，已经被 W3C 作为一种 Internet 标准采纳。

SOAP 是一个远程过程调用（RPC）协议，使用标准的 Internet 协议进行传输，同步调用时采用 HTTP 协议，异步调用时采用 SMTP 协议。由于可以在 HTTP 上运行，这使得 SOAP 在穿越防火墙进行操作方面优于 DCOM、RMI 和 IIOP，而在嵌入设备上实现 SOAP 也比开发一个 ORB 更简单。

SOAP 的主要设计目标是简单性和可扩充性。为了实现这两个目标，SOAP 中省略了在其他消息系统和分布式对象系统中常见的一些特性，如无用存储单元收集、消息批处理等。

SOAP 没有定义一种编程模型或实现，而是定义了一个模块化的包装模型，并在模块内定义了编码数据的编码机制。这使得 SOAP 可以在从消息传递系统到远程过程调用的任何系统中应用。

SOAP 由以下 4 个部分组成。

- 一个 SOAP 封皮（Envelope）：定义了描述消息所包含信息的框架结构，即消息中包含什么信息、由谁来处理以及是必需的或可选的。
- 一组 SOAP 编码规则（Encoding rules）：定义了一个串行化机制，用于交换应用定义的数据类型的实例。SOAP 编码的类型使用简单的标量类型和复合类型，如结构和数组，这些类型以 XML 文档元素的形式表现。XML Schema 规范中定义的数据类型以及这些数据类型的派生类型都可以直接用作 SOAP 元素。
- SOAP RPC 表示：定义如何表示远程过程调用和响应。SOAP 的设计目标之一是用 XML 的可扩展性和灵活性封装 RPC 功能，在 SOAP 1.2 中详细定义了 RPC 和响应的统一表示，将对一个方法的调用和响应作为结构来建模，结构中包含了返回值，或者还可能包括传入的参数。
- SOAP 绑定（binding）：定义如何使用底层传输协议进行 SOAP 消息的交换。虽然 SOAP 本身可以和多种协议结合使用，但 SOAP 1.2 中只描述了在 HTTP 中的使用。SOAP 和 HTTP 绑定可以同时使用 SOAP 的形式方法与分散的灵活性以及 HTTP 丰富的特性集。在 HTTP 中使用 SOAP 并不意味着 SOAP 覆盖了 HTTP 现有的语义，而是继承了 HTTP 的语义。

SOAP 消息是用 XML 编码的文档，它由封皮、消息头和消息体 3 个部分组成。

- SOAP 封皮（包 SOAP Envelope）：是描述 SOAP 消息的 XML 文档的顶点元素。
- SOAP 消息头（SOAP Header）：提供了一种灵活的机制对 SOAP 消息以分散的、模块化的方式进行扩充，而通信的各方（包括 SOAP 发送者、SOAP 接收者及 SOAP 中介）不必预先知道。需要注意的是，SOAP 消息头是可选的。
- SOAP 消息体（SOAP Body）：定义了一个简单的机制来交换要发送给最终 SOAP 接收者的消息中的必要信息，是这些信息的容器。典型的使用是编组 RPC 调用和 SOAP 错误报告。

SOAP 消息是单方向的，从一个 SOAP 发送者（sender）到一个 SOAP 接收者（receiver）。但单独的消息通常可以被组合在一起形成其他消息机制。例如，SOAP 通过在 HTTP 请求中提供一个 SOAP 请求消息和在 HTTP 响应中提供一个 SOAP 响应消息实现 HTTP 的请求/响应消息模型。SOAP 消息交换模型要求接收到一个 SOAP 消息的应用程序执行下列操作：

- 识别 SOAP 消息中意图供给本应用的部分，本应用可以作为 SOAP 中介，将消息的其他部分传递给另外的应用。
- 检验 SOAP 消息中指定的所有必须处理的部分，并进行相应的处理。
- 如果 SOAP 应用不是消息的最终目的地，它应该在删除所有自己消耗的部分后将消息转发给消息要供给的下一个应用。

SOAP 只是一种包装和绑定调用一个 Web 服务所需信息的方式，Web 服务也可以使用其他的编码技术调用。另外，SOAP 本身没有严格地归入 Web 服务，SOAP 可以作为一种



对任何类型的远程对象或过程的访问机制使用，也可以只是一个简单的消息传递机制。

除了 SOAP 以外，W3C 创建的 XMLP 工作组还建立了 XMLP 协议（Extensible Markup Language Protocol，XMLP）。XMLP 是类似于 SOAP 的 XML 消息协议，包括封皮、对象串行化方式、HTTP 传输绑定以及进行远程过程调用的方式几个部分。甚至有人认为 XMLP 将逐步取代 SOAP。

（3）WSDL（Web Services Description Language，Web 服务描述语言）

Web 服务的目标之一是允许应用程序以标准的方式在两个或多个同等的服务之间进行选择，因为有时应用可以由作为支持网络的服务而实现的构件构造而成，甚至可以从这些服务中进行动态选择。服务描述层定义了为程序提供足够信息所需的描述机制，使程序能够根据一定的准则选择服务，如服务的质量、安全性、可靠性等。

Web 服务的接口由基于 XML 的 WSDL 定义，它提供了应用访问指定的 Web 服务所必需的全部信息，如描述服务提供了什么功能、服务位于何处以及服务如何调用等。

WSDL 以 XML 格式描述网络服务，将服务描述为在包含面向过程或面向文档信息的消息上进行操作的一组端点。操作和消息是抽象描述的，然后绑定到具体的网络协议和消息格式以定义一个端点。相关的具体端点被组合成为抽象端点（服务）。WSDL 是可扩展的，允许描述任何端点和消息，而不考虑通信使用的消息格式或网络协议。

WSDL 使用下面的元素定义网络服务。

- 类型（Types）：用某种类型系统的数据类型定义的容器。WSDL 并没有引入新的类型定义语言，而将 XSD 作为自己的标准类型系统，并允许通过可扩展性使用其他的类型定义语言。
- 消息（Message）：对要传送的数据的一个抽象定义。
- 操作（Operation）：对服务支持的动作的抽象描述。
- 端口类型（Port Type）：一个或多个端点支持的操作的一个抽象集合。
- 绑定（Binding）：对特定端口类型的一个具体协议和数据格式规格。
- 端口（Port）：一个单独的端点，由一个绑定和一个网络地址组合在一起定义。
- 服务（Service）：一组相关的端点的集合。

一个 Web 服务由一组端口定义，而端口由绑定到一个具体协议和数据格式规范的一组抽象操作和消息定义。操作和消息的抽象是为了使它们可以复用和绑定到不同的协议和数据格式，如 SOAP、HTTP GET/POST 或 MIME。

在 WSDL 中，端点和消息的抽象定义是和它们的具体网络配置和数据格式绑定相分离的；另外，WSDL 定义了一个公共的绑定机制，用于将特定的协议、数据格式或结构连接到抽象的消息、操作或端点，这些都允许对抽象定义的复用。

（4）UDDI（Universal Description，Discovery，and Integration，统一描述、发现和集成）

UDDI 是一套基于 Web 的、分布式的、为 Web 服务提供的信息注册中心的实现标准规范，同时也包含一组使企业能将自身提供的 Web 服务注册，以使别的企业能够发现的访问协议的实现标准。UDDI 为表示 XML 中商业服务描述定义了一个数据结构标准，提供了更高层次的商业信息，以补充 WSDL 中的说明。UDDI 定义了如下 4 种基本的结构。

- 商业实体（Business entity）：描述商业信息，如名称、类型等。
- 商业服务（Business service）：已发布的 Web 服务的集合。
- 绑定模板（Binding template）：访问信息，如 URL。
- 技术规范（TModel）：对服务类型的技术规格说明，如接口定义、消息格式、消息协议和安全协议等。

在进行一个 Web 服务调用之前，必须先找到具有所需服务的企业，发现调用接口和语义，然后编写或配置自己的软件，以便与服务合作。UDDI 的核心部件是 UDDI 商业注册，它用一个 XML 文档来描述企业及其提供的 Web 服务。UDDI 商业注册是一个基于 SOAP 的 Web 服务，提供企业用于将它们的服务发布到注册中心的接口。注册中心是分布式的，彼此之间不断进行复制操作。

Web 服务基本上是机器到机器的通信，为了有效工作，这种体系结构必须具有进行基于 Web 的应用和业务过程集成的有效工具。UDDI 商业注册中心包含以下 3 类信息，企业可以通过这些信息发现一个 Web 服务。

- 白页：包括企业的名称、地址、联系方式和企业标识，并允许其他公司按照名称查找目录。
- 黄页：包括基于标准分类法的行业类别。
- 绿页：包括关于该企业所提供的 Web 服务的技术信息，其形式可能是一些指向文件或 URL 的指针，而这些文件或 URL 是为服务发现机制服务的。绿页还允许注册的公司之间使用 XML 进行连接，提供了业务过程自动化的关键机制。

UDDI 规范提供了编程接口，允许商业注册一个 Web 服务，以及查找指定 Web 服务的注册。一旦想要的 Web 服务被确定，将提供一个指向 WSDL 文档所在位置的指针。编程接口分为查询 API 和发布 API 两个逻辑部分。查询 API 又分为两个部分：一部分用来构造搜索和浏览 UDDI 注册信息的程序，另一部分在 Web 服务出现错误时使用。发布 API 可以用来创建各种类型的工具，以直接与 UDDI 注册中心进行交互，便于企业技术人员管理发布信息。

UDDI 规范包含了对基于 Web 的 UDDI 商业注册中心可以实施的整套共享操作。一般来说，程序或程序员通过 UDDI 商业注册中心来获得 Web 服务的位置及其技术信息。对于程序员来说，是对自己的系统实现准备，以使自己的系统能和那些 Web 服务实现访问兼容，或是描述自己的 Web 服务，从而能让别人使用。从商业层次上来说，UDDI 商业注册中心可以被用于核查某个合作伙伴是否拥有特定的 Web 服务的调用接口，或是找出在某行业中能提供某种类型服务的公司，并确定某合作伙伴的 Web 服务的技术描述及交互时所需的技术细节。

UDDI 是完全可选的，也就是说，具有 Web 服务的公司，如果只是想对有限的人员或设备提供特定功能，就不需要对外发布它们的服务。

1.3.2 Web 服务技术优势

Web 提出了面向服务的分布式计算框架，具有松散耦合、与平台无关、易于集成等优



点，为 Internet 上的分布式应用提供了有效的支持，业界众多的厂商也都投入到其研究和开发中。Web 服务技术优势表现在如下几个方面。

（1）平台无关性

Web 服务与具体的操作系统无关，与软件平台也无关。例如，Windows 的 Web 服务能够与 UNIX 下的 Web 服务方便地相互通信，同在 Windows 操作系统下的.NET 平台与 J2EE 平台之间同样能够容易地交互信息。在这以前从未有任何技术能做到这一步。

（2）松散耦合性

Web 服务是部署在网络上的对象，具有自然良好的封装。其中任何一个组件发生改变时，不需要修改其他组件来体现这种变化。它们之间的变化对对方来说是透明的。

（3）基于文本的简单性和自描性

Web 服务以 XML 技术为基础，而 XML 是基于文本的，并且具有自我描述的能力。

（4）可集成性

无论 Web 服务建立在何种软件平台上，用何种语言编写，都可以与其他 Web 服务实现当前环境下的最高的可集成性。

Web 服务的突出优点还在于它能够在完全不同的平台之间具有互操作性，通过遍布全球的 Internet 实现应用程序之间的远程过程调用。

1.3.3　Web 服务技术的研究方向及发展趋势

Web 服务作为一种新的技术应运而生，已经被业界称为继 PC 和 Internet 之后，计算机 IT 技术的第三次革命。Web 服务的出现，使得可以在现有各种异构平台的基础上构建一个通用的、与平台无关、与语言无关的技术层，各种不同平台之上的应用依靠这个技术层来实施彼此的连接和集成。

1．研究方向

（1）服务发布和发现

Web 服务发现是 Web 服务系统架构中的一个重要部分。Web 服务可能具有不同的内容、形式和复杂程度，如何对 Web 服务进行描述和组织，使请求者能够基于概率或语义约束的模糊匹配进行查找，实现服务发现的高效性、自动化和智能化，是 Web 服务研究的一个重要内容。

（2）服务组合

在业务处理中，通常需要按照一定粒度将多个 Web 服务根据特定的应用背景和需求进行合理的组合，实现完整的业务逻辑。人们在该领域进行了研究，如 WSFL、XLAN、GBPEL4WS 等。

（3）性能优化

XML 和 SOAP 使 Web 服务具有更好的开放性，但 XML 的解析和传输等使 Web 服务的性能与 CORBA 等调用相比，存在数量级上的差距，这将直接影响 Web 服务应用和推广。

业界提出了 SOAP 消息传输优化方案、XML 二进制优化建议等，因此如何对 XML 解析、SOAP 消息传输进行优化，是提高 SOAP 可用性的一个重要方面。

（4）安全性

为了保证 Internet 上 Web 应用的安全，防止信息被非法访问和修改，需要采用安全控制或信息加密等手段。现有的安全技术，如数字签名、XML 加密标准、访问控制技术等，在一定程度上解决了特定的安全问题，但如何实现 Web 服务安全保护的自动化，保证不同粒度和级别的数据机密性、完整和可用性仍然是一个重要的研究问题。WS2Security、WS2Trust 等规范提供的一个框架级别的安全标准还需要在应用中进一步验证。

（5）事务机制

Web 服务提供了 Internet 上应用和信息的集成，为了保证 Web 应用协同工作并保持一致，得到可靠的结果和输出，Web 应用需要有事务处理的机制。与传统事务相比，Web 服务中事务机制的特点为：事务的执行周期可能很长；Web 事务比传统事务更松散、更灵活、更复杂，并不严格地遵循传统事务 ACID 原则；事务参与者可能分布在网络中不同位置、不同平台上；服务组合中需要事务机制来保证其协调工作。目前，Web 服务支持的事务模型主要是 Business Transaction、WSDL Transaction 和 Activity Service，其策略通常是扩展已存在的事务处理技术，其实效性仍然有待进一步研究。

2. 发展趋势

Web 服务技术在各研究方向中还存在一些问题，例如，如何对 Web 服务进行描述和组织，使请求者能够基于概率或语义约束的模糊匹配进行查找，实现服务发现的高效性、自动化和智能化；对于与组合服务相关的各服务组件和基本服务，如何定义它们之间的逻辑及时序关系，以实现复杂 Web 服务执行的自动化；如何实现服务组件和基本服务之间的动态交互、协调及状态保持，以保证 Web 服务执行的有序性；如何保持语义信息及如何验证和测试组合 Web 服务，以确保 Web 服务执行结果的正确性；如何实现 Web 服务安全保护的自动化，以保证不同粒度和级别的数据机密性、完整和可用性等，这些问题都是 Web 服务技术在研究中需要解决的问题，也就成为 Web 服务技术下一步的发展趋势。特别是由于传统的 Web 服务技术缺乏机器可理解的语义，限制了 Web 服务的自动化，如何使 Web 信息为机器所理解并自动处理成为 Web 发展的趋势。

Web 服务与网格融合是未来发展趋势，网格计算（Grid Computing）使人们能够轻而易举地为一些大型科研任务创建和使用动态、分布式、高性能的计算环境。这些在以前是不可能实现的，因为其开展起来所付出的代价很高，如高能物理数据分析、气候建模、宇宙观测、实时遥感数据处理和虚拟现实等。同时它也能够应用在商业计算领域，如联机分析处理、数据挖掘和商业智能等。此外，它还能够广泛地应用在电子商务、电子政务等领域。网格计算使人们能够共享计算、存储、数据和应用资源。这种计算模式是利用互联网把分散在不同地理位置的电脑组织成一个“虚拟的终极电脑”，其中每台参加计算的电脑就是个“节点”，而整个计算是由成千上万个“节点”组成的“一张网格”。网格有别于 Web 的基本特征就在于服务的形式。如今，Web 要创建应用环境，仍要依靠研发人员按照 Web 协议研发，而网格是在更高层次上对这些应用提供的一种服务形式。因此，将来的应用系统所



基于的平台应是网格所提供的基本服务。而这种服务的本身，又会不断动态地加入到网格中，使得网格服务的内容不断丰富。由 IBM、Sun 和 Microsoft 倡导的全球 Grid 论坛（Global Grid Forum），把目标锁定在把网格计算技术和 Web 服务计算结合起来提供商务应用服务上，从而将网格计算技术从科学计算领域引入到商务应用领域，并引发了 IBM、Sun 和 Microsoft 采取行动，将网格计算和 Web 服务相结合，实现一种使业务交易在分布于 Internet 服务器上运行的技术。

OGSA 是在 Globus 网格计算工具包和 Web 服务技术融合的基础上提出的一套规范和标准。它与服务器版 Java、Web 服务连同商业数据库紧密集成，实现了网格计算在商务领域的广泛应用。OGSA 吸纳了许多 Web 服务标准，如 Web 服务描述语言（WSDL）、简单对象访问协议（SOAP）、目录访问协议（LDAP）和 Web 服务探测（WSInspection）等，这些标准用于定位、调度计算资源，并确保它们的安全。

1.4　Web 技术

本节简要介绍与 Web 相关联的技术，如 Web 数据库、ASP.NET、JSP、Ajax、语义 Web 和 Web 3D 等内容。

1.4.1　Web 数据库

众所周知，Internet 网络是全球最大的计算机通信网，人们通过 Web 可以浏览各个国家和地区网上的各种类型的信息资源，如静态图像、文本、数据、视频和音频。其实，Web 系统是一个大型的分布式超媒体信息数据库，其已经成为 Internet 中最流行、最主要的信息服务方式。用户能够在 Internet 上浏览、查询和共享建立在 Web 服务器所有站点上的超媒体信息。数据库厂家和 Web 公司也纷纷推出各自的产品和中间件支持 Web 技术与数据库管理系统（DBMS）的融合，将两者取长补短，发挥各自的优势，使用户可以在 Web 浏览器上方便地检索数据库的内容。可见，Web 数据库管理系统是基于 Web 模式的 DBMS 信息服务，该系统充分发挥了 DBMS 高效的数据存储和管理能力，以浏览器/服务器（B/S）模式为平台，为 Internet 用户提供方便、快捷、内容丰富的数据服务。因此，Web 数据库管理系统成为 Internet 和 Intranet 提供的核心服务，也为 Internet 上的电子商务提供技术支持。

通过 Web 访问数据库有以下 3 个优点：

- 无须开发数据库前端软件，通过浏览器就可以访问后台数据库。它具有界面统一、培训费用低、实用方便等优势。
- 开发者只需学习 HTML 一种语言就可以开发基于 Web 的应用系统。
- Web 服务器书写的 HTML 文档可以被所有平台的浏览器所浏览，实现了跨平台开发。因为各种操作系统上都有现成的浏览器可供使用。

Web 数据库应用系统通过在 Web 服务器端提供中间件来连接 Web 服务器和数据库服务

器。中间件负责管理 Web 服务器和数据库服务器之间的通信并提供应用程序服务，它能够直接调用外部程序或脚本代码来访问数据库，因此可以提供与数据库相关的动态 HTML 页面，或执行用户查询，并将查询结果格式化成 HTML 页面，通过 Web 服务器返回给 Web 浏览器。

常用的访问数据库驱动的方式有通用网关接口 CGI、应用程序接口 API、ODBC、JDBC 等多种渠道。下面分别介绍这些驱动技术。

（1）CGI（通用网关接口）访问技术

CGI 是 Web 服务器运行时外部程序的规范，按照 CGI 编写的程序可以扩展服务器的功能，完成服务器本身不能完成的工作，外部程序执行时间可以生成 HTML 文档，并将文档返回 Web 服务器。CGI 应用程序能够与浏览器进行交互作用，还可以通过数据库的 API 与数据库服务器等外部数据源进行通信，如一个 CGI 程序可以从数据库服务器中获取数据，然后格式化为 HTML 文档后发送给浏览器，也可以将从浏览器获得的数据放到数据库中。几乎所有的服务器软件都支持 CGI，开发人员可以使用任何一种 Web 服务器内置语言编写 CGI，其中包括流行的 C、C++、VB 和 Delphi 等。

从体系结构上来看，用户通过 Web 浏览器输入查询信息，浏览器通过 HTTP 协议向 Web 服务器发出带有查询信息的请求，Web 服务器按照 CGI 协议激活外部 CGI 程序，由该程序向 DBMS 发出 SQL 请求，并将结果转化为 HTML 后返回给 Web 服务器，再由 Web 服务器返回给 Web 浏览器。这种结构体现了客户/服务器方式的 3 层模型，其中，Web 服务器和 CGI 程序实际起到了 HTML 和 SQL 转换的网关的作用。CGI 访问数据库的具体操作过程为：

① 分析 CGI 数据。
② 打开与 DBMS 的连接。
③ 发送 SQL 请求并得到结果。
④ 将结果转化为 HTML。
⑤ 关闭 DBMS 的连接。
⑥ 将 HTML 结果返回给 Web 服务器。

基于 CGI 技术访问数据库的主要缺点如下：

- 客户端与后端数据库服务器通信必须通过 Web 服务器，且 Web 服务器要进行数据与 HTML 文档的互相转换，当多个用户同时发出请求时，必然在 Web 服务器形成信息和发布瓶颈。
- CGI 应用程序每次运行都需打开和关闭数据库连接，效率低，操作费时。
- CGI 应用程序不能由多个客户机请求共享，即使新请求到来时 CGI 程序正在运行，也会启动另一个 CGI 应用程序。随着并行请求的数量增加，服务器上将生成越来越多的进程。为每个请求都生成进程既费时又需要大量内存，影响了资源的使用效率，导致性能降低并增加了等待时间。
- 由于 SQL 与 HTML 差异很大，所以 CGI 程序中的转换代码编写繁琐，维护困难。
- 安全性差，缺少用户访问控制，对数据库难以设置安全访问权限。
- HTTP 协议是无状态且没有常连接的协议，DBMS 事务的提交与否无法得到验证，不能构造 Web 上的 OLTP 应用。



（2）应用程序接口 API 访问技术

为了克服 CGI 的局限性，出现的另一种中间件解决方案是基于服务器扩展 API 的结构。与 CGI 相比，应用程序 API 与 Web 服务器结合得更加紧密，占用的系统资源也少得多，但运行效率却大大提高，同时还提供更好的保护和安全性。

应用程序 AP 一般作为一个 DLL 提供，是驻留在 Web 服务器中的程序代码，其扩展 Web 服务器的功能与 CGI 相同。Web 开发人员不仅可以用 API 解决 CGI 可以解决的一切问题，而且能够进一步解决基于不同 Web 应用程序的特殊请求。各种 API 与其相应的 Web 服务器紧密结合，其初始开发目标服务器的运行性能进一步发掘、提高。用 API 开发的程序比用 CGI 开发的程序在性能上提高了很多，但开发 API 程序比开发 CGI 程序要复杂得多。API 应用程序需要一些编程方面的专门知识，如多线程、进程同步、直接协议编程以及错误处理等。目前主要的 WebAPI 有 Microsoft 公司的 ISAPI、Netscape 公司的 NSAPI 和 OReily 公司的 WSAPI 等。使用 ISPAI 开发的程序性能要优于用 CGI 开发的程序，这主要是因为 ISAPI 应用程序是一些与 Web 服务器软件处于同一地址空间的 DLL，因此所有的 HTTP 服务器进程能够直接利用各种资源，这显然比调用不在同一地址空间的 CGI 程序语句要占用更少的系统时间。而 NSAPI 与 ISAPI 一样，其他为 Web 开发人员定制了 Netscape Web 服务器基本服务的功能。开发人员利用 NSAPI 可以开发与 Web 服务器的接口以及与数据库服务器等外部资源的接口。

虽然基于服务器扩展 API 的结构可以方便、灵活地实现各种功能，连接所有支持 32 位 ODBC 的数据库系统，但这种结构也有缺陷，例如，各种 API 之间兼容性很差，缺乏统一的标准来管理这些接口；开发 API 应用程序也要比开发 CGI 应用复杂得多；这些 API 只能工作在专用 Web 服务器和操作系统上。

（3）ODBC 访问技术

ODBC（Open Database Connectivity，开放数据库互连）是微软公司开放服务结构（Windows Open Services Architecture，WOSA）中有关数据库的一个组成部分，它建立了一组规范，并提供了一组对数据库访问的标准 API（应用程序编程接口），这些 API 利用 SQL 来完成其大部分任务。ODBC 本身也提供了对 SQL 语言的支持，用户可以直接将 SQL 语句送给 ODBC。

ODBC 使用层次的方法来管理数据库，在数据库通信结构的每一层，对可能出现依赖数据库产品自身特性的地方，ODBC 都引入一个公共接口以解决潜在的不一致性，从而很好地解决了基于数据库系统应用程序的相对独立性，这也是 ODBC 一经推出就获得巨大成功的重要原因之一。

一个基于 ODBC 的应用程序对数据库的操作不依赖任何 DBMS，不直接与 DBMS 打交道，所有的数据库操作由对应的 DBMS 的 ODBC 驱动程序完成。不论是 FoxPro、Access、MySQL 还是 SQL Server、Oracle 数据库，均可用 ODBC 进行访问。应用程序要访问一个数据库，首先必须用 ODBC 管理器注册一个数据源，管理器根据数据源提供的数据库位置、数据库类型及 ODBC 驱动程序等信息，建立起 ODBC 与具体数据库的联系。这样，只要应用程序将数据源名提供给 ODBC，ODBC 就能建立起与相应数据库的连接。由此可见，ODBC 的最大优点是能以统一的方式处理所有的数据库。

在 ODBC 中，ODBC API 不能直接访问数据库，必须通过驱动程序管理器与数据库交换信息。驱动程序管理器负责将应用程序对 ODBC API 的调用传递给正确的驱动程序，而

驱动程序在执行完相应的操作后，将结果通过驱动程序管理器返回给应用程序。

（4）JDBC 访问技术

JDBC 是由 JavaSoft 公司指定的、用于执行 SQL 语句的 Java 应用程序接口 API，由 Java 语言编写的类和接口组成。Java 是一种面向对象、多线程与平台无关的编程语言，具有极强的可移植性、安全性和强健性。JDBC 是一种规范，能为开发者提供标准的数据库访问类和接口，能够方便地向任何关系数据库发送 SQL 语句。同时，JDBC 是一个支持基本 SQL 功能的低层应用程序接口，但实际上也支持高层的数据库访问工具及 API。所有这些工作都建立在 X/Open SQL CLI 基础上。JDBC 的主要任务是定义一个自然的 Java 接口来与 X/Open CLI 中定义的抽象层和概念连接。JDBC 的两种主要接口分别面向应用程序的开发人员的 JDBC API 和面向驱动程序低层的 JDBC Driver API。JDBC 完成的工作是建立与数据库的连接；发送 SQL 语句；返回数据结果给 Web 浏览器。

采用 JDBC 技术，在 Java Applet 中访问数据库的优点在于：直接访问数据库，不再需要 Web 数据库的介入，从而避开了 CGI 方法的一些局限性；用户访问控制可以由数据库服务器本地的安全机制来解决，提高了安全性；JDBC 是支持基本 SQL 功能的一个通用低层的应用程序接口，在不同的数据库功能的层次上提供了一个统一的用户界面，为跨平台、跨数据库系统进行直接的 Web 访问提供了方案，从而克服了 API 方法一些缺陷；同时，可以方便地实现与用户的交互，提供丰富的图形功能和声音、视频等多媒体信息功能。

1.4.2 ASP.NET 技术

ASP.NET 是 Microsoft 公司推出的新一代建立动态 Web 应用程序的开发平台，是一种建立动态 Web 应用程序的新技术。它是.NET 框架的一部分，可以使用任何.NET 兼容的语言（如 Visual Basic、C#）编写 ASP.NET 应用程序。当建立 Web 页面时，可以使用 ASP.NET 服务器端控件来建立常用的 UI（用户界面）元素，并对它们进行编程来完成一般的任务，这可以把程序开发人员的工作效率提升到与其他技术都无法比拟的程度。

1. ASP.NET 发展历程

ASP.NET 1.0 于 2000 年正式发布，并于 2003 年升级为 1.1 版本。ASP.NET 1.1 发布之后更加激发了 Web 应用程序开发人员对 ASP.NET 的兴趣，对网络技术有着巨大的推动作用。本着“减少 70%代码”的目标，微软公司在 2005 年 11 月又发布了 ASP.NET 2.0。ASP.NET 2.0 的发布是.NET 技术走向成熟的标志，它在使用上增加了方便、实用的新特性，使 Web 开发人员能够更加快捷、方便地开发 Web 应用程序，不但执行效率大幅度提高，对代码的控制也做得更好，以高安全性、易管理性和高扩展性等特点著称。微软还推出了 ASP.NET 3.5 版本，使网络程序开发更倾向于智能开发，运行起来更像 Windows 下的应用程序一样流畅。

2. ASP.NET 的优点

（1）高效的运行性能

ASP.NET 应用程序采用页面脱离代码技术，即前台页面代码保存到.aspx 文件，后台代



码保存到.cs 文件，这样当编译程序将代码编译为.dll 文件，ASP.NET 在服务器上运行时，可以直接运行编译好的.dll 文件，并且 ASP.NET 采用缓存机制，从而提高运行 ASP.NET 的性能。

（2）简易性和灵活性

很多 ASP.NET 功能都可以扩展，这样可以轻松地将自定义功能集成到应用程序中。例如，ASP.NET 提供程序模型，为不同数据源提供插入支持。

（3）可管理性

ASP.NET 中包含的新增功能使得管理宿主环境变得更加简单，从而为宿主主体创建了更多增值的机会。

（4）生产效率

使用新增的 ASP.NET 服务器控件和包含新增功能的现有控件，可以轻松、快捷地创建 ASP.NET 网页和应用程序。新增内容（如成员资格、个性化和主题）可以提供系统级的功能，此类功能通常会要求开发人员进行大量的编写代码工作。新增数据控件、无代码绑定和智能数据显示控件已经解决了核心开发方案。

1.4.3 JSP 技术

JSP（Java Server Pages）技术是由 Sun 公司发布的、用于开发动态 Web 应用的一项技术。它以其简单易学、跨平台的特性，在众多动态 Web 应用程序设计语言中异军突起，在短短几年中已经形成了一套完整的规范，并广泛地应用于电子商务等各个领域中。在国内，JSP 现在也得到了比较广泛的重视，越来越多的动态网站已采用该技术。

1. JSP 的运行原理

在一个 JSP 文件第一次被请求时，JSP 引擎把该 JSP 文件转换成为一个 Servlet，而这个引擎本身也是一个 Servlet。JSP 的运行过程如下：

（1）JSP 引擎先把该 JSP 文件转换成一个 Java 源文件（Servlet），在转换时如果发现 JSP 文件有任何语法错误，转换过程将中断，并向服务端和客户端输出出错信息。

（2）如果转换成功，JSP 引擎用 javac 把该 Java 源文件编译成相应的 class 文件。

（3）创建一个该 Servlet（JSP 页面的转换结果）的实例，该 Servlet 的 jspInit()方法被执行，jspInit()方法在 Servlet 的生命周期中只被执行一次。

（4）jspService()方法被调用来处理客户端的请求。对每一个请求，JSP 引擎创建一个新的线程来处理该请求。如果有多个客户端同时请求该 JSP 文件，则 JSP 引擎会创建多个线程，每个客户端请求对应一个线程。以多线程方式执行可以大大降低对系统的资源需求，提高系统的并发量及响应时间。但也应该注意多线程的编程限制，由于该 Servlet 始终驻于内存，所以响应是非常快的。

（5）如果 JSP 文件被修改了，则服务器将会根据设置决定是否对该文件重新编译，如果需要重新编译，则将编译结果取代内存中的 Servlet，并继续上述处理过程。

（6）虽然 JSP 效率很高，但在第一次调用时，由于需要转换和编译，所以会有一些轻微的延迟。此外，在任何时候，如果系统资源不足，JSP 引擎将以某种不确定的方式将 Servlet 从内存中移去。当这种情况发生时，jspDestroy()方法首先被调用。

（7）然后 Servlet 实例便被标记加入“垃圾收集”处理。可在 jspInit()中进行一些初始化工作，如建立与数据库的连接或建立网络连接，从配置文件中取一些参数等，以在 jspDestory()中释放相应的资源。

2. JSP 技术特点

（1）简便性和有效性

JSP 动态网页的编写与一般的静态 HTML 的网页编写是十分相似的。只是在原来的 HTML 网页中加入一些 JSP 专有的标签或脚本程序（此项不是必需的）。这样，一个熟悉 HTML 网页编写的设计人员可以很容易地进行 JSP 网页的开发，而且开发人员完全可以不自己编写脚本程序，而只是通过 JSP 独有的标签使用别人已写好的部件来实现动态网页的编写。这样，一个不熟悉脚本语言的网页开发者，完全可以利用 JSP 做出漂亮的动态网页，而这在其他的动态网页开发中是不可实现的。

（2）程序的独立性

JSP 是 Java API 家族的一部分，它拥有一般的 Java 程序的跨平台的特性。换句话说，就是拥有程序的对平台的独立性，即 Write once，Run anywhere!。

（3）程序的兼容性

JSP 中的动态内容可以各种形式进行显示，所以它可以为各种客户提供服务，即从使用 HTML/DHTML 的浏览器，到使用 WML 的各种手提无线设备（例如，移动电话和个人数字设备 PDA），再到使用 XML 的 B2B 应用，都可以使用 JSP 的动态页面。

（4）程序的可重用性

在 JSP 页面中可以不直接将脚本程序嵌入，而只是将动态的交互部分作为一个部件加以引用。这样，一旦这样的一个部件写好，它可以为多个程序重复引用，实现了程序的可重用性。现在，大量的标准 JavaBeans 程序库就是一个很好的例证。JSP 技术为创建显示动态生成内容的 Web 页面提供了一个简捷的方法。JSP 技术的设计目的是使得构造基于 Web 的应用程序更加容易和快捷，而这些应用程序能够与各种 Web 服务器、应用服务器、浏览器和开发工具共同工作。

1.4.4 Ajax 技术

基于 XML 的异步 JavaScript，简称 Ajax（Asynchronous JavaScript and XML），是多种技术的综合，它使用 XHTML 和 CSS 标准化呈现，使用 DOM 实现动态显示和交互，使用 XML 和 XSTL 进行数据交换与处理，使用 XMLHttpRequest 对象进行异步数据读取，使用 JavaScript 绑定和处理所有数据。更重要的是，它打破了使用页面重载的惯例技术组合，可以说 Ajax 已成为 Web 开发的重要武器。这个术语源自描述从基于网页的 Web 应用到基于



数据的应用的转换。在基于数据的应用中，用户需求的数据，如联系人列表，可以从独立于实际网页的服务端取得并且可以被动态地写入网页中，为缓慢的 Web 应用体验着色，使之像桌面应用一样。许多重要的技术和 Ajax 开发模式可以从现有的知识中获取。例如，在一个发送请求到服务端的应用中，必须包含请求顺序、优先级、超时响应、错误处理及回调，其中许多元素已经在 Web 服务中包含了，就像现在的 SOAP。Ajax 开发人员拥有一个完整的系统架构知识。同时，随着技术的成熟还会有许多地方需要改进，特别是 UI 部分的易用性。

综合各种变化的技术和强耦合的客户服务端环境，Ajax 提出了一种新的开发方式。Ajax 开发人员必须理解传统的 MVC 架构，这限制了应用层次之间的边界。同时，开发人员还需要考虑 CS 环境的外部和使用 Ajax 技术来重定型 MVC 边界。最重要的是，Ajax 开发人员必须禁止以页面集合的方式来考虑 Web 应用，而需要将其认为是单个页面。一旦 UI 设计与服务架构之间的范围被严格区分开后，开发人员就需要更新和变化的技术集合了。

Ajax 开发与传统的 C/S 开发有很大的不同，这些不同引入了新的编程问题，最大的问题在于易用性。由于 Ajax 依赖浏览器的 JavaScript 和 XML，浏览器的兼容性和支持的标准也变得和 JavaScript 运行时的性能一样重要了。这些问题中的大部分来源于浏览器、服务器和技术的组合，因此必须理解如何才能最好地使用这些技术。随着 Ajax 迅速地引人注目，开发人员对这种技术的期待也迅速地增加。与任何新技术一样，Ajax 的发展也需要一整个开发工具/编程语言及相关技术系统来支撑。

（1）JavaScript 编程语言

JavaScript 是 Ajax 概念中最重要但最被忽视的一种 JavaScript 编程语言。JavaScript 是一种粘合剂，使 Ajax 应用的各部分集成在一起。在大部分时间，JavaScript 通常被服务端开发人员认为是一种企业级应用，不需要使用的东西应该尽力避免。在 Ajax 中，JavaScript 主要被用来传递用户界面上的数据到服务端并返回结果。XMLHttpRequest 对象用来响应通过 HTTP 传递的数据，一旦数据返回到客户端就可以立刻使用 DOM 将数据放到网页上。

（2）XMLHttpRequest 对象

XMLHttpRequest 对象在大部分浏览器上已经实现而且拥有一个简单的接口，允许数据从客户端传递到服务端，但并不会打断用户当前的操作。使用 XMLHttpRequest 传送的数据可以是任何格式，虽然从名字上建议是 XML 格式的数据。

（3）CSS

为了正确地浏览 Ajax 应用，CSS 是一种 Ajax 开发人员所需要的重要武器。CSS 提供了从内容中分离应用样式和设计的机制。虽然 CSS 在 Ajax 应用中扮演至关重要的角色，但它也是创建跨浏览器应用的一大阻碍，因为不同的浏览器厂商支持各种不同的 CSS 级别。

（4）服务器端

在服务端，Ajax 应用还是应建立在如 Java、.NET 和 PHP 语言基础上，并没有改变这个领域中的主要方式。

不同的 IDE 提供了对 JavaScript 支持的不同等级。来自 JetBrains 的 IntelliJ IDEA 是一个用来进行 JavaScript 开发的更好的 IDE，虽然许多开发人员也喜欢 Microsoft's Visual Studio 产品（允诺会在最新的版本中改善对 Ajax 的支持）。Eclipse 包含了两个免费的 JavaScript

编辑器插件和一个商业的来自 ActiveStat 的 Komodo IDE。

JavaScript 编程的最大问题为不同的浏览器对各种技术和标准的支持。构建一个运行在不同浏览器（如 IE 和火狐）的 JavaScript 脚本是一件困难的事情。因此，几种 Ajax JavaScript 框架或者生成基于服务端逻辑或标记库的 JavaScript，或者提供符合跨浏览器 Ajax 开发的客户端 JavaScript 库。一些流行的框架包括 Ajxa.Net、Backbase、Bitkraft、Django、DOJO、DWR、MochiKit、Prototype、Rico、Sajax、Sarissa 和 Script.aculo.us.等。

这些框架给开发人员更多的空间，使得他们不需要担心跨浏览器的问题。虽然这些框架提升了开发人员构建应用的能力，但由于厂商已经开发了更细节的用户界面的打包组件解决方案，因此在 Ajax 组件市场中需要考虑一些其他因素，例如，提供通用用户界面的组件（如组合框和数据栅格）的几个厂商，都可以被用来在应用中创建良好的通过类似电子数据表方式来查看和编辑数据的体验。但这些组件不仅封装了组件的用户界面，而且包括与服务端数据的通信方式，这些组件通常使用基于标记的方式来实现如 ASP.NET 或 JSP 控件。

总而言之，Ajax 开发人员必须尽快地跟进最新的技术并利用高产的工具集。另外，成功的 Ajax 开发人员还应留心使用者，以避免将任何问题扩大化，同时还应持续地创新，从而创建增强 Web 应用易用性的新方法。

1.4.5 语义 Web 服务技术

Web 之所有获得成功，是因为它以一种简便的方式（HTTP、HTML、URL、浏览器）将整个 Internet 上的文本、声音、图像等各种信息集合在一起，使用户能够很方便地搜索和浏览多媒体信息。如果 Web 能成为一个巨大的数据库，将人类易读的文档和机器可理解的数据以一种对人类和机器都很有效的方式链接起来，将会怎样？例如，搜索“老泰山”，如果是采用基于关键字的搜索，可能返回大量与泰山有关的页面，而这与用户的本意相差太远。若是采用语义 Web 技术就可以较好地处理这个问题，它能根据精确的概念、知识结构和推理规则进行搜索，从而得到与用户目标比较接近的结果，如返回的结果就一定是与“岳父”有关的信息。

Web 之父，W3C（World Wide Web Consortium）总裁 Tim Berners-Lee 及 W3C 的其他领导人最初在 2001 年 5 月提出了语义 Web 观点，并随后成为 W3C 工作的一个重点工作。Tim Berners-Lee 等人认为：“语义 Web 是对当前 Web 的扩展，其中信息被赋予了定义好的含义，能够更好地支持计算机和人类的协同。”语义 Web 通过在现有的 Web 上增加元数据，允许人类和机器以一种现有几乎不可能的方式发现和利用数据。Tim Berners-Lee 预测，语义 Web 将是一个能够理解人类语言的智能网络，可识别信息的意义，并对信息自动进行解释、交换和处理，它允许 Web 实体（软件代理、用户和程序）进行互操作、动态地发现和使用资源、提取知识并解决复杂的问题。

元数据是对 Web 内容进行了注解和描述，人们利用元数据来实现计算机的操作。语义 Web 分资源集层、基础服务集层和应用程序集 3 层模型，其含义如下。



（1）Web 资源集：拥有唯一的全球标识，由元数据以一种通用、共享的方式描述，并拥有一系列通过本体论推理判断新的元数据和识别的规则。

（2）基础服务集：提供基于元数据、本体论和语义搜索引擎的推理和查询，是对现有 Internet 服务（如 DNS、基于关键字的搜索）的极大改进。

（3）使用基础服务集开发高层应用程序集。

总之，语义 Web 就是能理解人类语言的智能网络，可以使人与计算机之间的交流变得很轻松。通过将 Web 内容的语法结构和语义以知识表示形式显式地表示出来，以实现与其他信息源共享，使得人之间、人和机器之间以及机器与机器之间能准确地相互理解，从而实现最大程度的互操作性。语义 Web 要能顺利工作，就要求计算机能结构化地组织信息和规则集合，以使计算机利用规则进行自动推理。

语义 Web 服务是在 Web 上的软件构件，它实现某种功能以满足某种需求。第一，所引用的 Ontology 支持自动推理；第二，智能 Agent 能够理解 Ontology 中的概念。要实现这两个前提需要基于描述逻辑的本体形式语言（如 OWL）和标准的、更高层次的本体（如基于 OWL 的框架 OWL-S）以使本体所表达的语义得到统一。OWL 是一种定义和实例化“Web 本体”的语言，OWL 提供了 3 种表达能力递增的子语言 OWL Lite、OWL DL 和 OWLFull，以分别用于特定的实现者和用户团体。OWL Lite 用于提供给那些只需要一个分类层次和简单约束的用户，OWL DL 支持那些需要最强表达能力的推理系统的用户，OWL Full 支持那些需要尽管没有可计算性保证，但有最强的表达能力和完全自由的 RDF 语法的用户。OWL-S 结构图如图 1-3 所示。

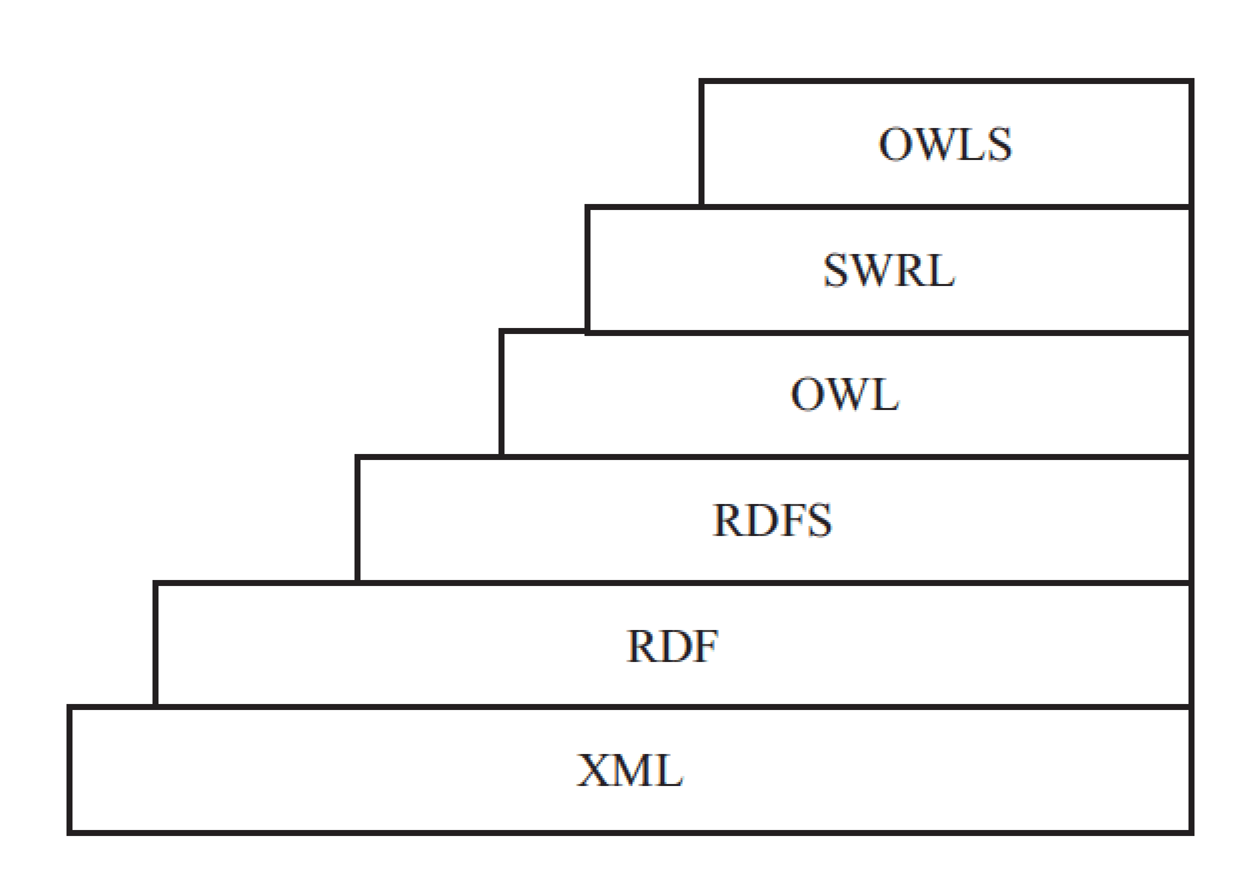


图 1-3　OWL-S 结构图

OWL-S 是其他描述技术的扩展，基于 XML 描述语言和其他的描述规则（RDF、RDFS、OWL、SWRL 等），是用本体来描述 Web 服务的标记语言。OWL-S 包括 3 个部分：服务简档、服务模型和服务基点，其中，服务简档主要用于 Web 服务的发现描述，它包括服务提供者的相关信息、服务的功能及其他属性、描述服务特性的其他信息（如服务的传输速度和可靠性）。

目前，语义 Web 服务研究的重点是元数据和本体论标准建模和实现的语言和技术的研究，如 XML Schema、RDF Schema、DAML+OIL、OWL，其中，RDF 是用来描述资源及其

之间关系的语言规范，它不仅是描述数据的框架，而且是表示数据的框架。RDF 的基本数据模型是一个三元组，其基本对象类型有资源、属性、陈述（又称声明）。每一个资源都具有属性，每一个资源通过唯一资源标识符 URI 来标识，它的属性由属性类型来标识，每个属性类型都有对应的属性值。属性类型表示出这些属性值与资源之间的关系。在 RDF 中，属性值要么是一些被公认的具有原子属性的事物，要么是其他的资源，而这些资源本身又拥有自身的属性，所有指向同一资源的陈述的集合称为该资源的一个描述。RDF 采用 XML 语言作为编码结构，两者相互兼容。其主要目标是定义一种机制来描述资源，使用 RDF 模型标识的元数据为语义 Web 提供有关语义信息的一致性控制。

RDF Schema 使用一种机器可以理解的体系来定义描述资源的词汇，其功能就像一个字典，可以将其理解为大纲或规范。RDFS 进一步定义了建模原语，其中，主要的类有 rdfs:Resourse、rdfs:Class 和 rdf:Property；主要属性有 rdf:type、rdfs:sub-ClassOf 和 rdfs:subPropertyOf；核心约束有 rdfs:domain、rdfs:ConstraintResource、rdfs:ConstraintProperty 和 rdfs:range。

本体（Ontology）是哲学上的概念，从哲学上讲，本体就是客观现实的抽象本质。在人工智能领域，本体是指给出构成相关领域词汇的基本术语和关系，以及利用这些术语和关系构成的规定这些词汇外延的规则的定义。现在计算机领域对本体最流行的定义就是概念模型的明确的规范说明。本体的目标就是获取相关领域的知识，提供对该领域知识的共同理解，确定该领域内共同认可的概念及其属性，并从不同层次的形式化模式上给出这些概念和概念之间相互关系的明确定义。人类、数据库和应用软件使用本体来共享领域知识（一个领域是指一个特定的学科范围或者知识范围，如医药、设备制造、房地产、汽车修理以及财务管理等）。本体既包括一个领域内的知识，也包括各种领域之间的知识，使用这种方式使知识被重用。本体描述语言有很多，包括 RDF(S)、OIL、DAML、DAML+OIL、OWL、SHOE 和 XOL 等。另外，开发本体的工具有很多，有 KAON OntoMat SOEP、OntoEdit、Ontolingua、Protégé 等。语义 Web 结构图如图 1-4 所示。

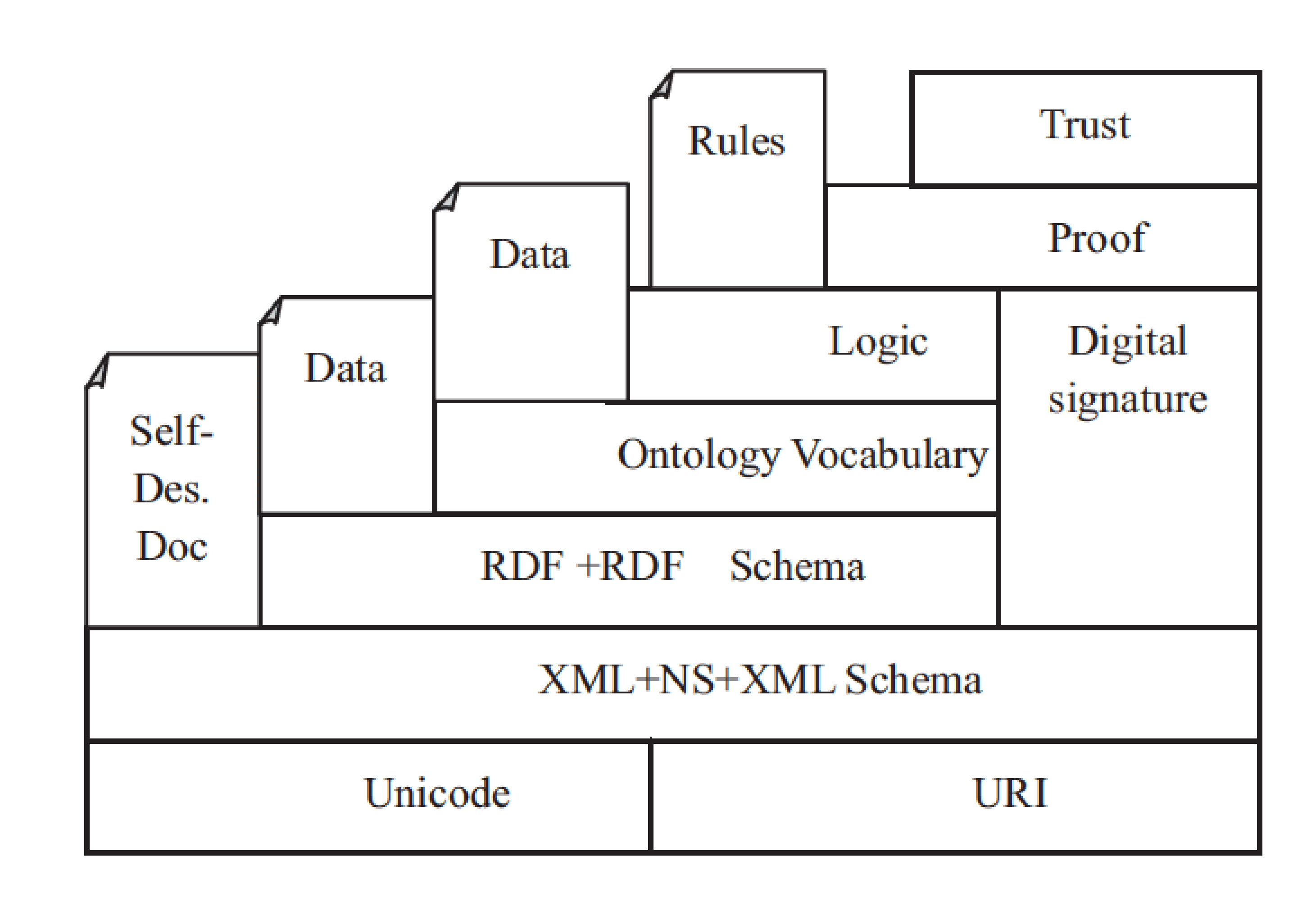


图 1-4　语义 Web 结构图



1.4.6　Web3D 技术

Web3D 技术是指基于 Internet 的、依靠软件技术来实现的桌面级虚拟现实技术，它是虚拟现实技术中的一种实现形式。Web3D 的出现最早可追溯到 VRML（Virtual Reality Modeling Language，即虚拟现实建模语言）。VRML 开始于 20 世纪 90 年代初期，1994 年 3 月在日内瓦召开的第一届 Web 大会上，首次正式提出了 VRML 这个名字。1998 年，VRML 组织更名为 Web3D 组织，同时制订了一个新的标准——Extensible 3D（X-3D）。2000 年的春天，Web3D 组织完成了 VRML 到 X-3D 的转换。X-3D 标准整合了正在迅速发展的 XML、Java、流媒体等先进技术，为 Web3D 提供了更强大、更高效的三维计算能力、渲染质量和传输速度。X-3D 标准的发布，为互联网 3D 图形的发展提供了广阔的应用前景，无论是小型的具有 3D 功能的 Web 客户端应用，还是高性能的广播级应用，都遵循 X-3D 标准，从而结束了当前互联网 3D 图形的混乱局面。在统一的 X-3D 基本框架下，保证了不同软件厂家开发的软件具有互操作性。目前，Web3D 技术已越来越多地应用在过程模拟、城市规划、商品展示和娱乐领域。Wed3D 技术将有广阔的发展前景，但仍然不可盲目乐观，它也面临着很多问题，如带宽、处理器速度等。现在的 Web3D 图形有几十种可供选择的技术和解决方案，多种文件格式和渲染引擎的存在是 Web3D 图形在互联网上应用的最大障碍。Web3D 技术主要是指建模技术、显示技术、三维场景中的交互技术。

1. 建模技术

虚拟现实技术的基础是三维复杂模型的实时建模与动态显示，该基础可以分为 3 类。一是基于几何模型的实时建模与动态显示；二是基于图像的实时建模与动态显示；三是三维扫描成型技术。在众多的 Web3D 开发工具中，Cult3D 是采用基于几何模型的实时建模与动态显示的技术，而 Apple 的 QTVR 则是采用基于图像的三维建模与动态显示技术。下面分别介绍这 3 种技术。

（1）基于几何模型的实时建模与动态显示技术

在计算机中建立起三维几何模型，一般均用多边形表示。在给定观察点和观察方向以后，使用计算机的硬件功能，实现消隐、光照及投影这一绘制的全过程，从而产生几何模型的图像。基于几何模型的建模与实时动态显示技术的主要优点是观察点和观察方向可以随意改变，不受限制，允许人们能够沉浸到仿真建模的环境中，充分发挥想象力，而不是只能从外部去观察建模结果。基于几何模型的实时建模与动态显示技术能够满足虚拟现实技术的 3I，即“沉浸”、“交互”和“想象”的要求。最常用的基于几何模型的建模软件是 3ds max 和 Maya。3ds max 是大多数 Web3D 软件所支持的，可以把它生成的模型导入使用。

（2）基于现场图像的 VR 建模技术

基于现场图像的 VR 建模就是通过摄像机对景物拍摄完毕后，自动获得所摄环境或物体的二维增强表象或三维模型。在建立三维场景时，选定某一观察点设置摄像机，每旋转一定的角度，便摄入一幅图像，并将其存储在计算机中；在此基础上实现图像的拼接，即将

物体空间中同一点在相邻图像中对应的像素点对准；对拼接好的图像实行切割及压缩存储，形成全景图。基于现场图像的虚拟现实建模有广泛的应用前景，它尤其适用于那些难于用几何模型的方法建立真实感模型的自然环境，以及需要真实重现环境原有风貌的应用。相对来说，基于图像的建模技术显然只能是对现实世界模型数据的一个采集，并不能够给 VR 设计者一个充分的、自由想象发挥的空间。

（3）三维扫描成型技术

三维扫描成型技术是用庞大的三维扫描仪来获取实物的三维信息，其优点是准确性高，但这样的扫描设备十分昂贵，对于 VR 的普通用户来说这似乎是遥不可及的事。

2．显示技术

显示技术就是把建立的三维模型描述转换成人们所见到的图像。因为在浏览 Web3D 文件时，一般都需要给用户安装一个支持 Web3D 的浏览器插件，这个对于初级用户来说也是一件麻烦的事情。但 Java3D 技术在这方面有很大优势，它不需要安装插件，在客户端用一个 Java 解释包来解释就行了。Microsoft 公司宣布，基于安全的理由，它不再支持 Java，操作系统 Windows XP 也没有内建 Java 虚拟机，所以如果在 Windows XP 使用 Java 3D 也必须安装 Java 虚拟机。其他 Web3D 软件是必须在客户端安装浏览器插件的。

3．交互技术

Web3D 实现的用户和场景之间的交互是相当丰富的，而在交互的场景中，实现用户和用户的交流也将成为可能，网络中关键是交互。总的来说，建立模型是用户首先要做的事情，也是相对困难的步骤；而显示是由软件通过计算机的运算完成的，用户不需要过问，只要选择显示质量能满足我们要求的技术就行了；交互功能的强弱由 Web3D 软件本身决定，但用户可以通过适当的编程来改善软件的不足。

下面介绍 4 款典型的 Web3D 开发工具软件。

（1）Viewpoint 工具

Viewpoint（Viewpoint Experience Technology）工具生成的文件格式非常小，三维多边形网格结构具有可伸缩（Scaleable）和流质传输（Steaming）等特性，使得它非常适合于在网络上应用。可以在它的 3D 数据下载的过程中看到一个由低精度的粗糙模型逐步转化为高精度模型的完整过程。在数据结构上，它分为两个部分，一是存储三维数据和贴图数据的.mts 文件，二是对场景参数和交互进行描述的基于 XML 的.mtx 文件。它具有一个纯软件的高质量实时渲染引擎，渲染效果接近真实而不需要任何的硬件加速设备。VET 可以和用户发生交互操作，通过鼠标或浏览器事件引发一段动画或一个状态的改变，从而动态地演示一个交互过程。VET 除了展示三维对象外还犹如一个能容纳各种技术的包容器，它可以把全景图像作为场景的背景，把 Flash 动画作为贴图使用。Viewpoint 的主要运用市场是作为物品展示的产品宣传和电子商务领域。许多著名的公司与电子商务网站使用了此技术作为产品展示，虽然不如 Cult3D 那样普及，但凭借着强大的功能还是赢得了不少用户的青睐，像 Fuji、Dell、Sony 等公司。



（2）Cult3D 工具

Cult3D 是一种崭新的 3D 网络技术，利用 Cult3D 技术可以做到档案小、3D 真实互动、跨平台运用，只要用鼠标在 3D 物件上直接拖动，就可以移动、旋转、放大缩小，还可以在 Cult3D 文件中加入音效和操作指引。对于窄带网的应用，Cult3D 也是最好的解决方案之一。Cult3D 的文件量非常小（20KB～200KB），却有近乎完美的三维质感表现，对于一般的浏览器只需安装一个插件即可浏览。和 Viewpoint 相比，Cult3D 在表观和交互上和 Viewpoint 相似，但 Cult3D 的内核基于 Java，它甚至可以嵌入 Java 类，利用 Java 来增强交互和扩展，Cult3D 的开发环境比 Viewpoint 人性化和条理化，开发效率也要高得多。如果要发布产品到网络上观看，Viewpoint 或 Cult3D 都是不错的选择，而 Shockwave3 适合开发三维在线游戏。

（3）Pulse3D 工具

Pulse3D 在娱乐游戏领域发展已经有好多年的历史，现在，Pulse 凭借其在游戏方面的开发经验把 3D 带到了网上，它瞄准的目标市场也是娱乐业。Pulse 提供了一个多媒体平台，囊括 2D/3D 图形、声音、文本、动画。Pulse 平台分为 3 个组件：Pulse Player、Pulse Producer 和 Pulse Creator。Pulse Player 也即播放器插件，除了为 IE 和 Netscape 提供的浏览器插件外，Pulse 还得到了 Apple 和 Real net work 的支持，在 Quicktime 和 RealPlayer 中已经包含了 Pulse 播放器。Pulse Producer 是用来在三维动画工具中输出 Pulse 所需数据的插件。常见的支持 3D 的软件有 3D studio max 和 Maya 的插件。能够输出到 Pulse 中的数据包括几何体网格、纹理、骨骼变形系统（支持 Character Studio）、Morph 网格变形动画、关键帧动画、音轨信息和摄像机信息。Pulse 还支持从 Vrml 和 BioVision 的输入。Pulse Creator 是 Pulse 总的组装平台。导入 Pulse Producer 生成的数据后，Pulse Creator 进行加入交互性、打光、压缩、流传输和缓存等功能操作。

（4）Plasma 工具

Plasma 是 3ds max 的 Web3D 版本，其简洁的界面、直观的用法、强大的 Havoc 引擎，从各种角度来说都是一款相当不错的软件。而 Plasma 支持 Flash、Shockwave 和 VRML 的输出，对于大部分 3D 设计师来说，这些功能已经很足够了。但是，也有不少人认为，Plasma 有点像是专门为 Shockwave 设计的建模工具，应用范围大大缩小了。而且 Plasma 的内容输出到 Shockwave 以后，固然能够表现出不错的质量，但是在 Flash 里面却并非如此，这似乎与注重写实感的 Web3D 项目开发用途有些不符。另外，它在支持 VRML 输出方面的功能比起 3ds max 或者其他软件来说并不占优势。Plasma 可以说是世界上最早的专门为 2D、3D Web 用户设计的三维建模、动画和渲染软件。作为 3D 建模工具，它完全继承了 3ds max 强大的建模功能，而且支持 Web Rendering（Flash Renderer）和 Exporting Tool，另外，它还整合了 Macromedia 公司的 Flash、Shockwave 3D 等设计工具和文件格式。从这些现象看来，Discreet 推出 Plasma 的一个很大的目标就是，通过让平面设计师掌握 3D 工具，从而更快地生成 Web3D 内容。Plasma 的主要功能和特征是可以转换为 Shockwave 3D 文件的，Plasma 文件可以输出成 Web3D 文件——Shockwave 3D Scene Export，而且还可以导入到 Director8.5。此外，Plasma 还可以输出为*.AL（Illustrator 文件）、*.DXF（AutoCAD 文件）和*.VRL（VRML 文件）3 种格式。Flash 动画制作可以说是 Plasma 最重要的功能之一。Plasma 有两种渲染方式，一种是 3ds max 中 Bitmap 方式的 Scanline 渲染方式，另外一种是矢量方式的 Flash 渲染

方式。这样，以前 Flash 用户需要经过长时间手动操作才能完成的建模过程就可以通过 Plasma 轻松完成了，而且能够节省大量的时间和费用。Flash 渲染方式不支持纹理，所以，渲染后的画面有明显的漫画风格。

1.5 小　结

本章着重介绍了关于 Web 的基本概念和技术，它们分别是 Web 定义、Web 工作模式（B/S）、Web 服务优势及发展趋势、通过 CGI、API、ODBC、JDBC 驱动方式访问 Web 数据库以及 ASP.NET、JSP、Ajax、语义 Web、Web3D 等技术知识。

1.6 思 考 题

1．Web 一词包含几层含义？
2．常见的 Web 服务器有哪几种？
3．Web 工作模式基于什么结构？
4．通过 Web 访问数据库有哪些驱动方式？
5．什么是语义 Web？
6．画出语义 Web 的分层结构。
7．Web3D 开发软件有哪些？各自的特点是什么？

第 2 章　HTML 及 XML 基础

Web 网页是用 HTML 和 XML 标记语言书写的具有特定格式的文档，那么标记语言是什么？本章将重点对标记语言的发展过程，特别是 HTML 及 XML 的概念进行详细、深入的介绍。

2.1　标记语言的发展历程

2.1.1　SGML

在 20 世纪 80 年代，为了方便数据的交换和控制，IBM 公司的内部人员开始寻求文档结构化的标准方法，提出在各文档之间共享一些相似的属性，如字体大小和版面。IBM 设计了一种文档系统，通过文档中辅加一种标签，从而可以标识文档中的每种元素。这样文档的显示和打印可能更少或更多地依赖特殊的硬件，不过这样的系统需要不同的计算机系统提供专门的软件来显示和打印文档。IBM 把自己这种标识语言称作通用标记语言（Generaized Markup Language），即 GML。

与此同时，其他机构也开发了类似这样的技术，但各自所有，互不兼容，没有统一的标准。GML 出现以后，IBM 没在其上做太多工作，直到 1986 年，国际标准化组织（ISO）认为 IBM 提出的通用标记语言这个概念很好，才发布了为生成标准化文档而定义的标记语言标准（ISO8879），并称之为标准通用标记语言（Standard Generalized Markup Language，SGML），即标准通用标记语言。

SGML 是一项强大和灵活的技术，因而不可避免地带来很大的复杂性和处理开销。

2.1.2　HTML

SGML 定义了许多不同类型的文档，HTML 即超文本标记语言（Hypertext Markup Language），定义了超文本文档的 SGML 的子集。人们习惯使用术语 HTML 表示超文本文档本身（属于一种特殊类型的 SGML 文档）和用以产生超文本文档的标记语言。用 HTML 编写的超文本文档称为 HTML 文档，它能独立于各种操作系统平台（如 UNIX、Windows 等）。自 1990 年以来，HTML 就一直被用作 World Wide Web 上的信息标记语言，用于描述 Homepage 的格式设计和它与 WWW 上其他 Homepage 的连接信息。虽然 HTML 语言描述了文档的结构格式，但并不能精确地定义文档信息必须如何显示和排列，而只是建议 Web

浏览器（如 Mosiac、Netscape 等）应该如何显示和排列这些信息，最终在用户面前的显示结果取决于 Web 浏览器本身的显示风格及其对标记的解释能力，这就是为什么同一文档在不同的浏览器中展示的效果会不一样。

2.1.3　XHTML

XHTML 是一种增强了的 HTML，它的可扩展性和灵活性将适应未来网络应用更多的需求。自从 1990 年 Tim Berners-Lee 设计出 HTTP/HTML 并创立 Web 后，万维网和超文本标记语言 HTML 取得了巨大成功，得到了十分广泛的应用。但是由于 HTML 的结构与外观混杂在一起，且内容往往被形式所掩盖，使数据的管理和信息的检索难以进行；加上微软和网景公司的浏览器各自的扩展互不兼容，给网页的设计与维护带来很大的困难。网页发布迫切需要一种兼容通用、内容与形式彻底分离且容易扩展的新网页发布的标记语言。基于以上原因，以 Berners-Lee 为首的 W3C（World Wide Web Consortium，万维网协会）于 1998 年推出了可扩展标记语言 XML（1.0: 1998.2.10），并于 2000 年用 XML 重写 HTML，推出了基于 XML 的新网页发布标记语言 XHTML（Extensible HyperText Markup Language，可扩展超文本标记语言）。

2.1.4　XML

XML 是 Extensible Markup Language 的缩写，意为可扩展的标记语言。XML 并不是一种新语言，它是 SGML 的一个子集，SGML 是一种非常通用的标记语言，但是非常难掌握。HTML 是 SGML 的子集，但由于 HTML 预定义的标记集合的限制使它失去了灵活性。XML 是元标记语言（Meta-Markup Language），可以用来创建特定领域的语言，而且 XML 是自解释语言，XML 的好处是易于掌握，易于理解。XML 首先创建于 1996 年，随后迅速发展起来，1998 年 2 月成为 W3C 标准。XML 文档的语法和语义定义由 W3C 负责。

2.1.5　DHTML

首先来看网站页面的一个重要特性——当浏览者将鼠标指针移动到页面导航条上时，会动态地弹出一个菜单，在该菜单中移动鼠标，所指向的菜单项变为红色显示；如果将鼠标指针移出菜单所在范围，则菜单自动隐藏；如果将鼠标指针移动到导航条上另外一个区域，则会弹出另外一个菜单，这种效果非常类似于 Windows 应用程序的特性，即通过图形化的界面为用户提供尽可能多的功能。实际上，采用这种方式可以使同一个页面中包含更多的信息，对于一个庞大的站点来说，这个特性非常有用。要实现这种效果，单纯依靠 HTML 和 JavaScript 已经无法实现，必须采用新的技术——这就是动态 HTML。所谓动态 HTML（Dynamic HTML），其实并不是一种新的语言，它只是 HTML、CSS 和客户端脚本的一种集成，即一个页面中包括 HTML+CSS+JavaScript（或其他客户端脚本），其中，CSS 和客户



端脚本是直接在页面上写而不是链接上相关文件。使用 DHTML 技术，可使网页设计者创建出能够与用户交互并包含动态内容的页面。实际上，DHTML 使网页设计者可以动态操纵网页上的所有元素，甚至是在这些页面被装载以后。利用 DHTML，网页设计者可以动态地隐藏或显示内容、修改样式定义、激活元素以及为元素定位。DHTML 还可使网页设计者在网页上显示外部信息，方法是将元素捆绑到外部数据源（如文件和数据库）上。所有这些功能均可用浏览器完成而无须请求 Web 服务器，也无须重新装载网页。这是因为一切功能都包含在 HTML 文件中，随着对网页的请求而一次性下载到浏览器端。

2.1.6　SHTML

SHTML 是 Server HTML 即服务器端的 HTML，当浏览网站时激活服务器上的 CGI 脚本程序而动态地自动生成。SHTML 是一个用于 SSI 技术的文件。SSI 是 Server Side Include 的缩写，通常称为“服务器端嵌入”，它是一种类似于 ASP 的基于服务器的网页制作技术。SHTML 和 HTML 格式差不多，SHTML 的页面可以使用 include 嵌入另外的 HTML 页面，这样可以把一个网站中的公用部分分离出来，然后再使用 include 将其嵌入到静态页面中，静态页面则不能。但由于安全性的问题和其他动态显示技术的出现，SHTML 的发展前景不是很好。

2.2　超文本标记语言 HTML

HTML 文档是放置了标记的文本文件，文件扩展名为.html 或.htm。

生成一个 HTML 文档主要有以下 3 种途径：

- 手工直接编写（如用文本编辑器或其他 HTML 的编辑工具）。
- 通过某些格式转换工具将现有的其他格式文档（如 Word 文档）转换成 HTML 文档。
- 由 Web 服务器（或称 HTTP 服务器）一方实时动态地生成。

HTML 的最高主版本为 4.0，HTML 将被 XHTML 所取代，但是 HTML 是 XHTML 的基础，因此，学习 HTML 还是非常必要的。

2.2.1　HTML 文件的页面结构

HTML 文档是由一系列的元素和标签组成的，元素名不区分大小写。HTML 用标签来规定元素的属性和它在文件中的位置。HTML 超文本文档分文档头和文档体两部分，在文档头中，对这个文档进行了一些必要的定义，文档体中才是要显示的各种文档信息。下面是一个最基本的 html 文档的代码。

例 2.1　用记事本编写如下代码：

```
<html>
<head>
<title> 一个简单的 HTML 示例 </ title >
</head>
<body>
<center>                      <h1>欢迎光临我的主页</h1>
<br>
 <hr>
  <font size= 7 color= red>
这是我第一次做主页
</font>
</center>
</body>
</html>
```

把该文件保存为以.html 为后缀的文件，双击该文件，就可以看到第一个主页的效果。在文档的最外层，文档中的所有文本和 html 标签都包含在<html>和</html>中，它标记该文档是以超文本标识语言（HTML）编写的。事实上，现在常用的 Web 浏览器都可以自动识别 HTML 文档，并不要求有标签，也不对该标签进行任何操作，但是为了使 HTML 文档能够适应不断变化的 Web 浏览器，还是应该养成不省略这对标签的良好习惯。

<head>和</head>是 HTML 文档的头部标签，在浏览器窗口中，头部信息是不显示在正文中的，在此标签中可以插入其他标记，用以说明文件的标题和整个文件的一些公共属性。若不需要头部信息则可省略此标记，良好的习惯是不省略。

<title>和</title>是嵌套在<head>头部标签中的，标签之间的文本是文档标题，它显示在浏览器窗口的标题栏。

<body>和</body>标记一般不省略，标签之间的文本是正文，是在浏览器中要显示的页面内容。

上面的这几对标签在文档中都是唯一的，head 标签和 body 标签是嵌套在 html 标签中的。

2.2.2　HTML 的基本标签

下面介绍 HTML 的基本标签。

（1）<html>和</html>

<html>标签放在 HTML 文档的最前面，用于标识 HTML 文档的开始；而</html>标签恰恰相反，它放在 HTML 文档的最后面，用于标识 HTML 文档的结束，两个标签必须成对使用，网页中所有其他的内容都要放在<html>和</html>之间。

（2）<head>和</head>

一个网页文档从总体上可分为头和主体两部分。<head>和</head>定义了 HTML 文档的头部分，必须是起始标签和结束标签成对使用。在此标签对之间可以使用<title >和</title>、



<script>和</script>等标签对，这些标签对都是描述 HTML 文档相关信息的标签对，<head>和</head>标签对之间的内容是不会在浏览器的文档窗口中显示出来的。

（3）<title>和</title>

使用过浏览器的人可能都会注意到浏览器窗口的标题栏上显示的文本信息，那些信息一般是网页的“主题”，要将网页的主题显示到浏览器的顶部其实很简单，只要在<title>和</title>标签对之间加入主题文本即可。注意，<title>和</title>标签对只能放在<head>和</head>标签对之间。

（4）<body>和</body>

<body>和</body>定义了 HTML 文档的主体部分，必须是起始标签与结束标签成对使用。在<body>和</body>之间放置的是实际要显示的文本内容和其他用于控制文本显示方式的标签，如<p>、</p>、<h1>、</h1>、
、</br>等，它们中间所定义的文本、图像等会在浏览器的窗口内显示出来。对于< body >标签，有以下一些主要属性。

- text：用于设定整个网页中的文字颜色。
- link：用于设定一般超链接文本的显示颜色。
- alink：用于设定鼠标移动到超链接上并按下鼠标时，超链接文本的显示颜色。
- vlink：用于设定访问过的超链接文本的显示颜色。
- background：用于设定背景墙纸所用的图像文件，可以是 GIF 或 JPEG 文件的绝对或相对路径。
- bgcolor：用于设定背景颜色，当已设定背景墙纸时，这个属性会失去作用，除非墙纸具有透明部分。
- leftmargin：设定网页显示画面与浏览器窗口左边沿的间隙，单位为像素。
- topmargin：设定网页显示画面与浏览器窗口上边沿的间隙，单位为像素。

<body>标签还有一些其他的公共属性，如 class、name、id、style 等，对这些属性确切的含义和作用，需要读者结合一些具体的实际应用加以理解。

2.2.3 超链接

例 2.2 建立一个超链接，代码如下：

```
<html>
<body>
<p>
<a href="../haut/html_xxxy/gongchengxi">这是一个链接</a>
</p>
<p>
<a href="http://www.haut.edu.cn/xxxy" target=_blank>河南工业大学 信息学院站点链接</a>
</p>
</body>
</html>
```

上面这个示例说明了如何在 HTML 文件中创建超链接。

例 2.3　建立一个图片超链接的代码如下：

```
<html>
<body>
<p>
将一张图片作为一个超链接，单击图片。
<a href="../haut/html_xxxy/gongchengxi.html"><img  src="../images/html_xxxy/123.jpg" ></a>
</p>
</body>
</html>
```

这个示例说明了如何将一个图片作为一个超链接，单击一个图片，即可链接到其他文件。

下面首先讲解 a 的用法。a 是 anchor 的缩写，<a>可以指向任何一个文件源、一个 HTML 网页、一张图片或一个影视文件等，其用法如下：

```
<a href="url">链接的显示文字</a>
```

单击<a>和当中的内容，即可打开一个链接文件，其属性如下。

- href 属性：指明链接文件的路径。如链接到 haut.edu.cn/xxxy 站点首页，就可以表示为：

```
<a href="http://www.haut.edu.cn/xxxy">河南工业大学 信息学院首页</a>
```

- target 属性：可以在一个新窗口里打开链接文件。如：

```
<a href="http://www.haut.edu.cn/html" target=_blank>河南工业大学 信息学院首页</a>
```

- title 属性：可以让鼠标悬停在超链接上时，显示该超链接的文字注释。如：

```
<a href="http://www.haut.edu.cn/xxxy" title = "河南工业大学 信息学院 WEB 基础教程中文站点">河南工业大学 信息学院网站</a>
```

如果希望注释多行显示，可以使用“
”作为换行符。如：

```
<a href="http://www.haut.edu.cn/xxxy" title = "河南工业大学 信息学院&#10;WEB 基础教程中文站点">河南工业大学 信息学院网站</a>
```

- name 属性：可以跳转到一个文件的指定部位。使用 name 属性，要设置一对，一是设定 name 的名称，二是设定一个 href 指向这个 name。

例 2.4　name 属性的用法。

```
<html>
<body>
<p>
<a href="#C6">参见第 6 章</a>
</p>
<p>
```



```
<a name="C1"><h2>第 1 章</h2></a>
<p>这是河南工业大学网站 信息学院-Web 教程中文站点。</p>
<a name="C2"><h2>第 2 章</h2></a>
<p>这是河南工业大学网站 信息学院-Web 教程中文站点。</p>
<a name="C3"><h2>第 3 章</h2></a>
<p>这是河南工业大学网站 信息学院-Web 教程中文站点。</p>
<a name="C4"><h2>第 4 章</h2></a>
<p>这是河南工业大学网站 信息学院-Web 教程中文站点。</p>
<a name="C5"><h2>第 5 章</h2></a>
<p>这是河南工业大学网站 信息学院-Web 教程中文站点。</p>
<a name="C6"><h2>第 6 章</h2></a>
</body>
</html>
```

name 属性通常用于创建一个大文件的章节目录（table of contents）。每个章节都建立一个超链接，放在文件的开始处，每个章节的开头都设置 name 属性。当用户单击某个章节的超链接时，这个章节的内容就显示在最上面。如果浏览器不能找到 name 指定的部分，则显示文章开头，不报错。在网站中，可能经常会看到“联系我们”超链接，单击该超链接，就会触发邮件客户端，如 Outlook Express，然后显示一个新建 mail 窗口。用<a>可以实现这样的功能，如：

```
<a href = "mailto:gongchengxi@haut.edu.cn">联系工程系</a>
```

2.2.4 列表

HTML 有 3 种列表形式：排序列表（Ordered List）、不排序列表（Unordered List）和定义列表（Definition List）。

（1）排序列表

在排序列表中，每个列表项前标有数字，表示顺序。排序列表由<ol>开始，每个列表项由<li>开始。

例 2.5 排序列表示例代码。

```
<html>
<body>
<h4>一个排序列表(Ordered List)：</h4>
<ol>
<li>河南工业大学 信息学院之网页课程</li>
<li>河南工业大学 信息学院之 Web 课程</li>
<li>河南工业大学 信息学院之 Java 课程</li>
</ol>
</body>
</html>
```

（2）不排序列表

不排序列表不用数字标记每个列表项，而采用一个符号标志每个列表项，如圆黑点。不排序列表由<ul>开始，每个列表项由<li>开始。

例 2.6　不排序列表的示例代码。

```
<html>
<body>
<h4>不排序列(Unordered List)：</h4>
<ul>
<li>河南工业大学 信息学院之网页课程</li>
<li>河南工业大学 信息学院之 Web 课程</li>
<li>河南工业大学 信息学院之 Java 课程</li>
</ul>
</body>
</html>
```

（3）定义列表

定义列表通常用于术语的定义。定义列表由<dl>开始。术语由<dt>开始，英文意为 definition term。术语的解释说明，由<dd>开始，<dd>和</dd>中的文字缩进显示。

例 2.7　定义列表示例。

```
<html>
<body>
<h4>定义列表(Definition List)：</h4>
<dl>
<dt>人</dt>
<dd>能够直立行走并会使用工具的高级动物，具备一定程度的高级智商。</dd>
<dt>黄种人</dt>
<dd>指皮肤棕黄色，多居住在亚洲</dd>
</dl>
</body>
</html>
```

2.2.5　表格

HTML 中用<table>标记表格。一个表格可以分成很多行（row），用<tr>表示；每行又可以分成很多单元格（cell），用<td>表示。

例 2.8　显示表格示例。

```
<html>
<body>
<p>
<head>表格所用到的标签：整个表格开始要用 table，每一行开始要用 tr，每一单元格开始要用 td。
```



```
</p>
<h4>只有一行（Row）一列（Column）的表格</h4>
<table border="1">
<tr>
<td>100</td>
</tr>
</table>
<h4>一行三列的表格</h4>
<table border="1">
<tr>
<td>100</td>
<td>200</td>
<td>300</td>
</tr>
</table>
<h4>两行三列的表格</h4>
<table border="1">
<tr>
<td>100</td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td>400</td>
<td>500</td>
<td>600</td>
</tr>
</table>
</body>
</html>
```

例 2.9　不显示边界的表格示例。

```
<html>
<body>
<h4>默认情况下，表格没有边界。</h4>
<table>
<tr>
<td>100</td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td>400</td>
<td>500</td>
<td>600</td>
```

```
</tr>
</table>
<h4>表格 Border 设为 0，也不显示边界：</h4>
<table border="0">
<tr>
<td>100</td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td>400</td>
<td>500</td>
<td>600</td>
</tr>
</table>
</body>
</html>
```

例 2.10　显示边界的表格示例。

```
<html>
<body>
<h4>表格的边界值设为 1：</h4>
<table border="1">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
<h4>表格的边界值设为 8，边界较粗：</h4>
<table border="8">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
<h4>表格的边界值设为 15，边界更粗：</h4>
<table border="15">
<tr>
```



```
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td></tr>
</table>
</body>
</html>
```

要显示表格的边界，可使用 border 这个属性。

例 2.11　有表头的表格示例。

```
<html>
<body>
<h4>有表头的表格：</h4>
<table border="1">
<tr>
<th>姓名</th>
<th>电话</th>
<th>传真</th>
</tr>
<tr>
<td>gongchengxi</td>
<td>555 77 854</td>
<td>555 77 855</td>
</tr>
</table>
<h4>竖直方向的表头：</h4>
<table border="1">
<tr>
<th>姓名</th>
<td>gongchengxi</td>
</tr>
<tr>
<th>电话</th>
<td>555 77 854</td>
</tr>
<tr>
<th>传真</th>
<td>555 77 855</td>
</tr>
</table>
</body>
</html>
```

表头用<th>来表示，表头的字是粗体显示的。

如果表格的单元格<td>和</td>之间没有内容，那么这个单元格的边界是不会显示出来的，尽管整个表格已设置边界值。要显示这个单元格的边界，可以插入一个“ ”空格符。

例 2.12　空的单元格示例。

```
<html>
<body>
<table border="1">
<tr>
<td>Some text</td>
<td>Some text</td>
</tr>
<tr>
<td></td>
<td>Some text</td>
</tr>
</table>
<p>
```

上面的表格中，有一个单元格里是没有任何内容的，这个空的单元格没有显示边界，虽然整个表格设了边界值。

```
</p>
<table border="1">
<tr>
<td>Some text</td>
<td>Some text</td>
</tr>
<tr>
<td> </td>
<td>Some text</td>
</tr>
</table>
<p>
```

通过上面的例子可以看出，给这个单元格加上一个空格符号之后，单元格的边界就显示出来了。注意，空格符要用“&bsp;”表示，“ ”是一个 HTML 特别字符，参见 HTML 特别字符（HTML Character Entities）。

```
</p>
</body>
</html>
```

例 2.13　有标题的表格示例。

```
<html>
<body>
```



```
<h4>
这个表格有标题：
</h4>
<table border="6">
<caption>表格标题</caption>
<tr>
<td>100</td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td>400</td>
<td>500</td>
<td>600</td>
</tr>
</table>
</body>
</html>
```

例 2.14　包含多列或多行的单元格示例。

```
<html>
<body>
<h4>用 colpsan 属性，设置包含多列的单元格：</h4>
<table border="1">
<tr>
<th>姓名</th>
<th colspan="2">联系方式</th>
</tr>
<tr>
<td>Bill Gates</td>
<td>555 77 854</td>
<td>555 77 855</td>
</tr>
</table>
<h4>用 rowspan 这个属性，设置包含多行的单元格：</h4>
<table border="1">
<tr>
<th>姓名</th>
<td>Bill Gates</td>
</tr>
<tr>
<th rowspan="2">联系方式</th>
<td>555 77 854</td>
</tr>
<tr>
```

```
<td>555 77 855</td>
</tr>
</table>
</body>
</html>
```

例 2.15　单元格中的内容示例。

```
<html>
<body>
<table border="1">
<tr>
<td>
<p>这是一段</p>
<p>这是另外一段。</p>
</td>
<td>这个单元格中包含了一个表格：
<table border="1">
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
</tr>
</table>
</td>
</tr>
<tr>
<td>这个单元格中包含了一张图片：
<img src = "../images/html_xxxy/mail.gif">
</td>
<td>HELLO</td>
</tr>
</table>
</body>
</html>
```

这个例子演示单元格<td>和</td>中的内容。单元格的内容可以是文字、图片、超链接、Form 和表格等。

例 2.16　单元格内容与单元格边界之间的距离示例。

```
<html>
<body>
<h4>没有 cellpadding 的表格：</h4>
```



```
<table border="1">
<tr>
<td>First</td>
<td>Row</td>
</tr>
<tr>
<td>Second</td>
<td>Row</td>
</tr>
</table>
<p>设置 cellpadding 属性，可以改变单元格内容和单元格边界之间的距离。</p>
<h4>以下是设置了 cellpadding 属性的表格：</h4>
<table border="1" cellpadding="10">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
<h4>cellpadding 属性值设置为 30 的表格：</h4>
<table border="1" cellpadding="30">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
</body>
</html>
```

这个示例说明如何用 cellpadding 属性设置单元格内容与单元格边界之间的距离。

例 2.17　单元格之间的距离示例。

```
<html>
<body>
<p>cellspacing 属性表示表格中单元格之间的距离。</p>
<h4>如果表格中没有设置 cellspacing 属性，默认情况下，cellspacing 值为 1。</h4>
<table border="1">
<tr>
<td>第一</td>
```

```
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
<h4>cellspacing 属性值设为 0 的表格：</h4>
<table border="1" cellspacing="0">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
<h4>cellspacing 属性值设为 20 的表格：</h4>
<table border="1" cellspacing="20">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
</body>
</html>
```

这个示例说明如何用 cellspacing 属性设置单元格之间的距离。

例 2.18　设置表格的背景颜色和背景图片示例。

```
<html>
<body>
<h4>给表格设置背景颜色：</h4>
<table border="1" bgcolor="red">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
```



```
</tr>
</table>
<h4>给表格加背景图片：</h4>
<table border="1" background="../images/html_tutorials/background.gif">
<tr>
<td>第一</td>
<td>行</td>
</tr>
<tr>
<td>第二</td>
<td>行</td>
</tr>
</table>
</body>
</html>
```

这个示例说明如何用 bgcolor 属性设置表格的背景颜色，如何用 background 属性为表格添加背景图片。

例 2.19　设置单元格的背景颜色和背景图片示例。

```
<html>
<body>
<h4>单元格背景色：</h4>
<table border="1">
<tr>
<td bgcolor="red">第一</td>
<td>行</td>
</tr>
<tr>
<td background="../images/html_tutorials/background.gif">第二</td>
<td>行</td>
</tr>
</table>
</body>
</html>
```

这个示例说明如何用 bgcolor 属性设置单元格的背景颜色，如何用 background 属性为单元格添加背景图片。

例 2.20　单元格内容的对齐方式示例。

```
<html>
<body>
<table width="350" border="1">
<tr>
<th align="center">分数</th>
<th align="center">期中考试</th>
```

```
<th align="center">期末考试</th>
</tr>
<tr>
<td align="left">芙蓉姐姐</td>
<td align="right">250.10</td>
<td align="right">50000.20</td>
</tr>
<tr>
<td align="left">程菊花</td>
<td align="right">1000.00</td>
<td align="right">5000.45</td>
</tr>
<tr>
<td align="left">周笔畅</td>
<td align="right">2000.40</td>
<td align="right">500.00</td>
</tr>
<tr>
<td align="left">黄新</td>
<td align="right">300.50</td>
<td align="right">800.65</td>
</tr>
</table>
</body>
</html>
```

这个示例说明如何用 align 属性设置单元格的对齐方式。

2.2.6 表单

表单（Form）是 HTML 的一个重要部分，主要用于采集和提交用户输入的信息。

例 2.21　让用户输入姓名的表单示例。

```
<form action="http://www.haut.edu.cn/xxxy/gongchengxi/ yourname.asp" method="get">
请输入你的姓名：
<input type="text" name="yourname">
<input type="submit" value="提交">
</form>
```

完整代码如下所示：

```
<html>
<head><title>输入用户姓名</title></head>
<body>
<form action="http://www.haut.edu.cn/xxxy/gongchengxi/ yourname.asp" method="get">
```



```
请输入你的姓名：
<input type="text" name="yourname">
<input type="submit" value="提交">
</form>
</body>
</html>
```

学习 HTML 表单最关键要掌握以下 3 个要点：

（1）表单控件（Form Controls）

通过表单的各种控件，用户可以输入文字信息、从选项中选择以及做提交的操作。如上面的示例中，“input type= "text"”就是一个表单控件，表示一个单行输入框。用户填入表单的信息总是需要程序来进行处理。

（2）表单中的 action

表单中的 action 就指明了处理表单信息的文件。如上面示例中的“http://www.haut.edu.cn/xxxy/gongchengxi/yourname.asp”。

（3）表单中的 method

表单中的 method 表示发送表单信息的方式。method 有两个值：get 和 post。get 方式是将表单控件的 name/value 信息经过编码之后，通过 URL 发送（可以在地址栏中看到），而 post 则将表单的内容通过 http 发送，在地址栏看不到表单的提交信息。那什么时候用 get，什么时候用 post 呢？一般是这样来判断的，如果只是为取得和显示数据，用 get；一旦涉及数据的保存和更新，建议用 post。

表单常用控件（Controls）如表 2-1 所示。

表 2-1　表单（Form）常用控件

表单控件（Form Controls）	说　明
input type="text"	单行文本输入框
input type="submit"	将表单（Form）中的信息提交给表单中 action 所指向的文件
input type="checkbox"	复选框
input type="radio"	单选按钮
select	下拉列表框
textArea	多行文本输入框
input type="password"	密码输入框（输入的文字用*表示）

2.2.7　框架

使用框架（Frame）可以在浏览器窗口同时显示多个网页。需要在每个 Frame 中设定一个网页，每个 Frame 中的网页相互独立。下面详细介绍。

（1）Frameset

<frameset>和</frameset>决定如何划分 Frame。<frameset>有 cols 属性和 rows 属性。使用 cols 属性，表示按列分布 Frame；使用 rows 属性，表示按行分布 Frame。

（2）Frame

可以用<frame>标签设定网页。<frame>有 src 属性，src 值就是网页的路径和文件名。下面的代码的作用是：将 Frameset 分成 2 列，第 1 列 25%，表示第 1 列的宽度是窗口宽度的 25%；第 2 列 75%，表示第 1 列的宽度是窗口宽度的 75%。第 1 列中显示 a.html，第 2 列中显示 b.html。

```
<frameset cols="25%,75%">
  <frame src="../asdocs/html_tutorials/Frame_a.html">
  <frame src="../asdocs/html_tutorials/Frame_b.html">
</frameset>
```

（3）Iframe

Iframe 是 Inline Frame 的意思，用<iframe>和</iframe>可以将 Frame 置于一个 HTML 文件内。

例 2.22　列分和行分 Frame 示例。

```
<html>
<frameset cols="25%,50%,25%">
<frame src="../asdocs/html_tutorials/Frame_a.html">
<frame src="../asdocs/html_tutorials/Frame_b.html">
<frame src="../asdocs/html_tutorials/Frame_c.html">
</frameset>
</html>
```

该示例显示如何在浏览器中同时显示 3 个网页，3 个网页是按列分布的。行分 Frame 的代码如下：

```
<html>
<frameset rows="25%,50%,25%">
<frame src="../asdocs/html_tutorials/Frame_a.html">
<frame src="../asdocs/html_tutorials/Frame_b.html">
<frame src="../asdocs/html_tutorials/Frame_c.html">
</frameset>
</html>
```

该示例显示如何在浏览器中同时显示 3 个网页，3 个网页是按行分布的。

例 2.23　混合 Frameset 示例。

```
<html>
<frameset rows="50%,50%">
<frame src="../asdocs/html_tutorials/Frame_a.html">
<frameset cols="25%,75%">
<frame src="../asdocs/html_tutorials/Frame_b.html">
<frame src="../asdocs/html_tutorials/Frame_c.html">
</frameset>
</frameset>
</html>
```



该示例既用到了 cols 属性，又用到了 rows 属性，将 Frame 进行灵活分布。Frameset 中的 noresize 属性示例代码如下：

```
<html>
<frameset rows="50%,50%">
<frame noresize="noresize" src="../asdocs/html_tutorials/Frame_a.html">
<frameset cols="25%,75%">
<frame noresize="noresize" src="../asdocs/html_tutorials/Frame_b.html">
<frame noresize="noresize" src="../asdocs/html_tutorials/Frame_c.html">
</frameset>
</frameset>
</html>
```

使用 noresize 属性可以确保 Frame 的大小。如果不使用 noresize 属性，可以通过用鼠标移动 Frame 的边界来改变 Frame 的大小，如果设置了 noresize 属性，就不能移动边界。

2.2.8　图像

用<img>标签可以在 HTML 中插入图片。其基本语法格式如下：

```
<img src="url">
```

其中，url 表示图片的路径和文件名。例如，url 可以是“images/logo/blabla_logo01.gif”，也可以是相对路径“../images/logo/blabla_logo01.gif”。

例 2.24　在 HTML 中插入图片示例。

```
<html>
<body>
<p>
河南工业大学 信息学院 Logo 图片：<img src="../images/html_xxxy123_logo.gif">
</p>
</body>
</html>
```

下面介绍图像标签的属性。

（1）alt 属性

alt 的英文全称为 alternate text。例如：

```
<img src = "../images/html_tutorials/smile.jpg" alt="smile face">
```

假使浏览器没有载入图片的功能，浏览器就会转而显示 alt 属性的值。其实，现在大多数浏览器都支持图片载入。此处介绍 alt 属性，是因为目前搜索引擎抓取工具无法识别图像中所含的文字，所以用 alt 属性写上图片的说明，便于搜索引擎抓取网页的内容。

例 2.25 alt 属性应用示例。

```
<html>
<body>
<p>将鼠标停留在图片上，可以看到 alt 属性中写的内容。</p>
<img src="../images/logo/123_logo.gif" alt="河南工业大学 信息学院的 Logo 标志" ><br>
<img src="../images/html_xxxy/frjj01.jpg" alt="信息学院：党员动员大会" >
</body>
</html>
```

（2）align 属性

使用 align 属性可以改变图片的垂直（居上、居中、居下）对齐方式和水平对齐方式（居左、居中、居右）。

例 2.26 align 属性应用示例。

```
<html>
<body>
<p>图片的垂直对齐方式：</p>
<p><img src="../images/logo/123_logo.gif" align="top">对齐方式:top</p>
<p><img src="../images/logo/123_logo.gif" align="middle">对齐方式:middle</p>
<p><img src="../images/logo/123_logo.gif" align="bottom">对齐方式:bottom</p>
<p>图片的水平对齐方式：</p>
<p><img src="../images/logo/123_logo.gif" align="Left">对齐方式:left</p>
<p><img src="../images/logo/123_logo.gif" align="center">对齐方式:center</p>
<p><img src="../images/logo/123_logo.gif" align="right">对齐方式:right</p>
</body>
</html>
```

（3）height 和 width 属性

在默认状况下，图片显示原有的大小。用 height 和 width 属性可以改变图片的大小。不过图片的大小一旦被改变，会相应放大或缩小，显示出来的结果可能会很难看。

例 2.27 height 和 width 属性应用示例。

```
<html>
<body>
<p>可以使用 height 和 width 属性来改变图片的大小。</p>
<p><img src="../images/html_xxxy/smile.jpg"></p>
<p><img src="../images/html_xxxy/smile.jpg" width="30" height="30"></p>
<p><img src="../images/html_xxxy/smile.jpg" width="100" height="80"></p>
<p><img src="../images/html_xxxy/smile.jpg" width="300" height="300"></p>
</body>
</html>
```

图片相对文字所占的字节数较多，如一个全屏的图片，即使经过压缩，也要占大约 50KB 字节，这相当于 25000 字的文本，因此浏览器载入图片比较费时，建议一个 HTML 文件中



不要包含过多的图片，否则会影响网页显示速度。

2.2.9　文本格式及其他

1. 文本格式

HTML 定义了一些文本格式的标签，例如，可以将字体变成粗体或者斜体。从下面的示例中，可以了解各种文本格式标签如何改变 HTML 文本的显示。常用的文本格式标签如表 2-2 所示。

表 2-2　文本格式标签说明

标　　签	说　　明
<b>	粗体 bold
<i>	斜体 italic
<del>	文字当中划线表示删除
<ins>	文字下划线表示插入
<sub>	下标
<sup>	上标
<blockquote>	缩进表示引用
<pre>	保留空格和换行
<code>	表示计算机代码，等宽字体

HTML 常用的格式代码如下：

```
<html>
<body>
<p><b>粗体用 b 表示。</b></p>
<p><i>斜体用 i 表示。</i></p>
<p><del>党校学习</del>这个词当中划线表示删除。</p>
<p><ins>暑期活动</ins>这个词下划线插入。</p>
<p>X<sub>2</sub>其中的 2 是下标</p>
<p>X<sup>2</sup>其中的 2 是上标</p>
<p><blockquote>好好学习，天天向上。这句话缩进表示引用</blockquote></p>
<pre>
这是预设（preformatted）文本，在 pre 标签中的文本保留空格和分行。
</pre>
<code>call getOrders</code>
<p>用 code 显示计算机代码，code 中显示的字符是等宽字符。</p>
</body>
</html>
```

上面程序用了很多标签，可以对比一下 HTML 的显示结果。

在 HTML 中用<p>和</p>划分段落。

```
<p>This is a paragraph</p>
<p>This is another paragraph</p>
```

简单的段落代码如下：

```
<html>
<body>
<p>这是第 1 段。</p>
<p>这是第 2 段。</p>
<p>这是第 3 段。</p>
<p>在 HTML 中，用 p 来定义段落。</p>
</body>
</html>
```

下面的代码介绍如何在 HTML 文件中定义正文标题。HTML 用<h1>到<h6>（从大到小）这几个标签来定义正文标题，每个正文标题自成一段。

```
<h1>This is a heading</h1>
<h2>This is a heading</h2>
<h3>This is a heading</h3>
<h4>This is a heading</h4>
<h5>This is a heading</h5>
<h6>This is a heading</h6>
```

通过使用
标签实现换行，可以在不新建段落的情况下换行。用<p>换行是个坏习惯，正确的是使用
，但
没有 Closing 标签，例如：

```
<p>This <br> is a para<br>graph with line breaks</p>
```

2．HTML 注释

在 HTML 文件中，可以写代码注释，用于解释说明代码，这样有助于日后更好地理解代码。注释可以写在<!--和-->之间。浏览器是忽略注释的，不会在 HTML 正文中看到注释内容，例如：

```
<!-- This is a comment -->
```

建议：HTML 文件会自动截去多余的空格。不管加多少空格，都被看作是一个空格。一个空行也被看作是一个空格。有些标签能够将文本自成一段，而不需要使用<p>和</p>来分段，如<h1>和</h1>之类的标题标签。

2.2.10 修饰字体

在 HTML 中，可以用 font 元素及其属性来设定字体的大小、颜色和字体风格。字体大小用字体大小属性（size）来设定。示例代码如下：



```
<html>
<head><title>字体大小 font size</title></head>
<body>
<p><font size="1">这段文字的字体大小值为 1。</font></p>
<p><font size="2">这段文字的字体大小值为 2。</font></p>
<p><font size="3">这段文字的字体大小值为 3。</font></p>
<p><font size="4">这段文字的字体大小值为 4。</font></p>
<p><font size="5">这段文字的字体大小值为 5。</font></p>
<p><font size="6">这段文字的字体大小值为 6。</font></p>
<p><font size="7">这段文字的字体大小值为 7。</font></p>
</body>
</html>
```

字体风格用 face 属性来设定。示例代码如下：

```
<html>
<head><title>字体风格 font face</title></head>
<body>
<p>以下第 1 段用的是 arial 字体，第 2 段用的是 courier 字体，第 3 段用的是 verdana 字体。</p>
<p><font face="arial">arial courier verdana</font></p>
<p><font face = "courier">arial courier verdana</font></p>
<p><font face="verdana">arial courier verdana</font></p>
</body>
</html>
```

字体颜色用颜色属性（color）来设定。颜色有两种表示方式，一种是用颜色名称表示，如 blue 表示蓝色；另一种是用十六进制的数值表示 RGB 的颜色值。R、G、B 分别代表 Red、Green、Blue，RGB 每个原色的最小值为 0，最大值为 255，如果换算成十六进制表示，就是（#00）、（#FF）。如白色的 RGB（255,255,255）用#FFFFFF 表示；黑色的 RGB（0,0,0）用#000000 表示。注意，在 W3C 制定的 HTML 4.0 标准中，只有 16 种颜色可以用颜色名称表示（aqua、black、blue、fuchsia、gray、green、lime、maroon、navy、olive、purple、red、silver、teal、white 和 yellow），其他的颜色都要用十六进制 RGB 颜色值表示。当然，现在的浏览器支持更多的颜色名称。但为保险起见，建议还是采用十六进制 RGB 颜色值来表示颜色，并且在值前加上符号“#”。

```
<html>
<head><title>字体颜色 font color</title></head>
<body>
<p>这段文字用的是默认的字体颜色。</p>
<p><font color="#FF0000">这段文字的字体颜色为红色。</font></p>
<p><font color="gray">这段文字的字体颜色为灰色。</font></p>
<p><font color="#33CC00">这段文字的字体颜色为绿色。</font></p>
</body>
</html>
```

注意：在 HTML 4 的标准中，已经不建议使用 font 及其属性来设定字体，而是用 CSS 样式来设定字体的大小、颜色和字体风格等。

2.2.11 网页设计

学习完上面 HTML 标记的基础知识之后，下面通过一个实际例子的制作（个人主页），依次掌握标签、字标题标签、文本、超链接、图片和表格等各类 HTML 标记的使用，在学习过程中要理论与实际相结合。

个人主页源代码如下：

```
<html>
<head><title>欢迎访问我的个人主页</title>
<style>
<!--
.a{
position:absolute;
left:8cm;
top:0cm;
z-index:2;
font-size:36px;
font-weight:bold;
color:yellow;
}
.b{
position:absolute;
left:8.05cm;
top:0.05cm;
z-index:1;
font-size:36px;
font-weight:bold;
color:yellow;
}
.c{
color:pink;
font-style:italic;
font-weight:bold;
font-size:20px;
}
.d{
color:pink;
font-style:normal;
font-weight:bold;
font-size:25px;
}
```



```
.e{
color:pink;
font-style:normal;
font-weight:bold;
font-size:18px;
}
-->
</style>
</head>
<body background="0.jpg">
<table frame="void" rules="none" bordercolor="purple">
<tr height="150">
<td colspan="3"><img src="1.gif"></td></tr>
<tr height="150">
<td ><img src="71.jpg"><div class="e"><a href="王菲-红豆.mp3">音频：</href></div>曾经以为过去的我们能忘了，但是，我们却忘不掉，总是不经意间在脑海中闪现。</td>
<td rowspan="3">
<embed src="巧虎(1).swf"  width="401 "height="304"  loop="-1"  controls>
<td><div class="e">hello!I'am honey</div></td>
</tr>
<tr height="150">
<td><img src="72.jpg"><div class="e"><a href="巧虎(1).swf"> 我的个人主页</href></div>
以前那么不经意的瞬间，现在却总是在脑海中盘旋，还记得那时你爱谈天我爱笑，怎么也没想到，现在却是相隔天涯海角，多么希望时光能够倒流，倒流到我们一块坐在麦堆上谈天说地。</td></td>
<td><div class="e">name:巧虎<br>weight:58kg</div></td>
</tr>
<tr height="150">
<td><img src="73.jpg"><div class="e"><a href="黄金矿工.swf">flash：</href></div>我知道我不是一个很好的记录者，但我比任何人都喜欢回首自己来时的路，我不断地回首、伫足，然而时光扔下我轰轰烈烈地向前奔去</td>
<td><div class="e">口头禅：听过了！我去玩了！
<br>酷照:</div><img src="74.jpg"></td>
</tr>
</table>
<br><br>
        <table><tr>
<td height="200"><a href="游戏界面.html"><img src="003.gif"></href></td>
<td height="200"><a href="我的家园.html"><img src="81.jpg"></href></td>
<td height="200"><a href="宝贝儿心情.html" ><img src="82.jpg"></href></td>
<td height="200"><a href="情感密码.html"><img src="83.jpg"></href></td>
<td height="200"><a href="九九乘法表档.html"><img src="002.gif"></href></td>
</tr>
<tr>
<td><div class="e"></div></td>
<td><div class="e"><a href="我的家园.html">偶的房产</div></href></td>
<td><div class="e"><a href="宝贝儿心情.html" >宝贝儿心情</div></href></td>
```

```
<td><div class="e"><a href="情感密码.html" >悄悄话</div></href></td>
<td><div class="e"></div></td>
</tr>
</table>
        
查看更多:<a href="felling for love.html">1</href> <a href="爱情 36 计.html">2</href> <a href="宝贝儿心情.html">3</href> <a href="看我 72 变.html">4</href> <a href="游戏界面.html" >5</href> <a href="情感密码.html">6</href> <a href="lovefoever.html">7</href>  
<br><br><br><br><br><br><br><br><br><br>访问过该网页的还有<br><br>
<table>
<tr><td><img src="3.jpg"></td>
<td><img src="90.jpg"></td>
<td><img src="93.jpg"></td>
<td><img src="94.jpg"></td>
<td><img src="96.jpg"></td>
<td><img src="5.jpg"></td>
<td><img src="91.jpg"></td>
<td><img src="95.jpg"></td>
<tr>
</table>
</body>
</html>
```

通过灵活使用以上各类标记，就能够建立一张类似于“个人主页”的精美网页。

2.3 扩展标记语言 XML

2.3.1 XML 基础

XML 是 Extensible Markup Language 的简写，是一种扩展性标识语言。那么该怎么来理解扩展性标识语言这个概念呢？

大家对 HTML 已经非常熟悉了，它就是一种标记语言，HTML 的全称为 Hypertext Markup Language，意为超文本标记语言。同时，HTML 中有很多标签，都是在 HTML 4.0 中规范和定义的，而 XML 中允许自己创建这样的标签，所以叫做可扩展性。XML 概念说明如下。

1. XML 并不是标记语言

XML 并不是标记语言，它只是用来创造标记语言(如 HTML)的元语言。XML 和 HTML 是不一样的，它的用途比 HTML 广泛得多，后面章节会详细介绍。



2．XML 并不是 HTML 的替代产品

XML 不是 HTML 的升级，它只是 HTML 的补充，为 HTML 扩展更多功能。今后将在较长的一段时间内继续使用 HTML。

3．不能用 XML 直接写网页

即便是包含了 XML 数据，依然要转换成 HTML 格式才能在浏览器上显示。下面是一段 XML 示例文档，用来表示本文的信息：

```
<myfile><br><br>
<title>XML Quick Start</title><br><br>
<author>ajie</author><br><br>
<email>ajie@aolhoo.com</email><br><br>
<date>20010115</date><br><br>
</myfile>
```

注意： 这段代码仅仅是代码，通过它可以初步感性地认识 XML，并不能实现什么具体应用；其中类似<title>、<author>的语句就是自己创建的标记(tags)，它们和 HTML 标记不一样，例如，这里的<title>是文章标题的意思，HTML 中的<title>是页面标题。

也许有读者会问有了 HTML，为什么还需要用 XML？那是因为现在网络应用越来越广泛，仅靠 HTML 单一文件类型来处理千变万化的文档和数据已经力不从心，而且 HTML 本身语法十分不严密，严重影响网络信息传送和共享。人们很早就已经开始探讨用什么方法来满足网络上各种应用的需要，使用 SGML 是可以的，但 SGML 太庞大，编程复杂，于是最终选择了“减肥”的 SGML——XML 作为下一代 Web 运用的数据传输和交互的工具。从以下 W3C 组织（XML 标准制定者）的说明中可以体会到使用 XML 的好处：XML 使得在网络上使用 SGML 语言更加“简单和直接”；简化了定义文件类型的过程，简化了编程和处理 SGML 文件的过程，简化了在 Web 上的传送和共享。具体体现在以下几个方面：

- XML 可以广泛地运用于 Web 的任何地方。
- XML 可以满足网络应用的需求。
- 使用 XML 使编程更加简单。
- XML 便于学习和创建。
- XML 代码清晰且便于阅读理解。

这些现在看起来还比较抽象，在后面的讲解中会慢慢体会到 XML 的强大优势。

2.3.2　XML 文档类型定义

XML 作为一门标记语言，它需要一种文档（即文档类型定义 DTD）来定义，DTD 可以看作是一类 XML 文档的模板，它定义了文档的逻辑结构，规定了 XML 文档中所使用的

元素及其属性、实体以及元素与实体之间的关系，它使得数据交流与共享得以正常进行，验证了数据的有效性。

DTD 可以是一个完全独立的文件，也可以在 XML 文件中直接设定。所以，DTD 分为外部 DTD（在 XML 文件中调用另外已经编辑好的 DTD）和内部 DTD（在 XML 文件中直接设定 DTD）两种。在一些有相互业务往来的公司，如果他们使用的电子文档是 XML 文档，那么就可以定义一个独立的 DTD 文档。每次交换和定义时都引用它来验证结构完整性和语法的合法性。例如下面一个 XML 文档：

```
<?xml version ="1.0" encoding ="GB2312" standalone="yes" ?>
  <学生名单>
    <学生>
      <学号>20081021</学号>
      <姓名>张三</姓名>
      <班级>计 08.2 班</班级>
    </学生>
    <学生>
      <学号>20081022</学号>
      <姓名>李四</姓名>
      <班级>计 08.2 班</班级>
    </学生>
  </学生名单>
```

它的一个 DTD 文档如下：

```
[1] <?xml version="1.0" encoding="GB2312" standalone="yes"?>
[2] <! DOCTYPE 学生名单[
[3] <!ELEMENT  学号(#PCDATA)>
[4] <!ELEMENT  姓名(#PCDATA)>
[5] <!ELEMENT  班级(#PCDATA)>
[6] <!ELEMENT  学生(学号,姓名,班级)>
[7] <!ELEMENT  学生名单(学生,学生)>
[8] ]>
```

第 2 行为 DTD 定义开始标记，学生名单为其根元素，第 3～7 行都是元素定义，第 8 行是结束标记。下面介绍元素声明的语法及注意事项。

1. 元素声明

元素声明的格式为：

<!ELEMENT 元素名称 元素的内容格式的定义>

- 基本元素声明：如“<!ELEMENT 学号 (#PCDATA)>”，元素名称后直接跟的是数据类型，则为基本元素。
- 复合元素声明：如“<!ELEMENT 学生 (学号,姓名,班级)>”，“学生”元素是复合元素，它包含学号、姓名和班级 3 个基本元素。其实，根元素是复合元素的一个



特例，所有的元素都直接或间接地包含在根元素中。

2．元素出现次数的控制

加“?”表示该元素可出现 0 次或 1 次；加“*”表示该元素可出现任意次；加“+”表示该元素至少要出现一次。如“<!ELEMENT 学生 (学号?,姓名,班级,爱好*)>”表示一个学生只有唯一的学号或者还未分配学号，即“学号”元素出现 0 次或 1 次，而爱好可以没有，也可以有好几方面的，所以用“*”(任意次）限定。选择性元素用“|”限定，如“<!ELEMENT 学生 (姓名,性别,(优秀|良好|中等))>”。

3．属性声明

属性声明的格式如下：

<!ATTLIST Element_name Attribute_name Type [Keyword] [Default_value]>

!ATTLIST 为定义属性的指令，后面是元素的名称、属性名称、属性值类型及默认值的关键字及默认值。如一个学生有性别及班级两个属性，可定义为“<!ATTLIST 学生 性别 CDATA "女" 班级 CDATA "计算机 03.2 班">”。必须赋值的属性要加 REQUIRED 关键字，如“<!ATTLIST 学生 性别 CDATA #REQUIRED "女" 班级 CDATA "计算机 03.2 班"> ”表示必须给出学生的性别值。可有可无的属性则用 IMPLIED 关键字，固定取值的属性用 FIXED 关键字。

4．属性的类型

在 XML 中共有 10 种属性类型，如表 2-3 所示。

表 2-3　XML 中的 10 种属性类型

类　型	含　义
CDATA	字符数据
Enumerated	可能的取值列表
ID	唯一的数字
IDREF	ID 类型属性的值
IDREFS	由空格分开的若干个 ID
ENTITY	实体
ENTITYS	若干个实体
NMTOKEN	XML 名称
NOTATION	DTD 中声明的注释名
NMTOKENS	多个 XML 名称

2.3.3　XML 数据的底层结构

XML 文档用于存储数据；Schema 或 DTD 用于验证数据，也就是结构化数据；DOM 是文档对象模型，作用是向 XML 文档中插入数据；XSLT 则负责数据的显示。

1. XML 数据的底层结构 DTD

文档类型定义 DTD（Document Type Definition）是 XML 结构文件的一种定义方式。DTD 定义了可用在文档中的元素、属性、实体以及它们之间的相互关系。建立一个 XML 文档的主要步骤如下：

（1）命名相关的信息项，把其映射为相应的元素或属性。

（2）确定 XML 文档的层次结构，即各元素之间的嵌套关系。

（3）根据层次结构构造 DTD。

（4）根据 DTD 编写相应的 XML 文档。

DTD 的语法结构包括元素定义和属性定义两部分，下面分别进行介绍。

1）元素定义

（1）元素声明

形式：<!ELEMENT name content >

说明：其中，name 为 XML 标记的名字；content 为 EMPTY 或 ANY，用于描述子元素的顺序和重复次数的内容模型。

（2）元素内容的类型

- EMPTY 类型：只有属性没有字符数据或子元素。
- ANY 类型：包含 DTD 定义的所有其他元素或已编译的字符数据。
- #PCDATA 类型：不包含其他子元素而只包含字符数据的元素。
- 子元素类型：包含一系列的子元素，子元素的内容模型用于指定某个元素可以包含哪些子元素。
- 混合类型：既包含子元素又包含已编译的字符数据。

（3）元素出现次数的指示符

- ?：元素可以出现 0 次或 1 次。
- *：元素可以不出现，或者出现一次或多次。
- +：元素必须至少出现一次，即可以出现一到多次。

2）属性定义

（1）属性声明

语法：<! ATTLIST Element_name Attribute_name Type Default_value >

说明：

- ATTLIST：用于定义元素所具有的属性。
- Element_name：元素名。
- Attribute_name：该元素所具有的属性。
- Type：属性的类型。
- Default_value：属性的默认值。

（2）属性的类型

- CDATA 类型：此属性的值只能是文本类型。
- 枚举属性类型：指定的文本串列表中的某个文本串。



- ID 属性类型：用于标识文档中的元素。
- IDREF/IDREFS 类型：用于引用同一文档中的另一元素的 ID 属性。
- NMTOKEN/NMTOKENS 属性类型：必须为一个有效的 XML 名称。
- ENTITY 和 ENTITYS 属性类型：用来引用文档中的不可解析的外部实体。
- NOTATION 类型：用于把属性值和 DTD 中的<! NOTATION>声明关联起来。

（3）属性的默认值

- REQUIRED：这个元素在使用时必须具有该属性。
- IMPLIED：该属性不是必须使用的。
- FIXED：在相应的 XML 文档中可以不用明确地指定该属性的值，如果明确指定属性值，则必须是定义时给出的默认值。

例 2.28　XML 文件代码示例。

```
<?xml version="1.0"?>
<!DOCTYPE message[
<!ELEMENT message (header,body,(signature|footer))>
<!ELEMENT header (date,from,to+,subject,banner?)>
<!ELEMENT body   (paragraph*)>
<!ELEMENT date   (date,month,year)>
<!ELEMENT paragraph (#PCDATA)>
<!ATTLIST paragraph size CDATA #REQUIRED>
<!ELEMENT signature (#PCDATA)>
<!ELEMENT footer   ANY>
<!ELEMENT day     (#PCDATA)>
<!ELEMENT month     (#PCDATA)>
<!ATTLIST month type (numberic|character) #REQUIRED>
<!ELEMENT year     (#PCDATA)>
<!ATTLIST year     format (numeric|character) "numeric" >
<!ELEMENT from   (#PCDATA)>
<!ELEMENT to (#PCDATA)>
<!ATTLIST to relationship CDATA #IMPLIED>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT banner   (#PCDATA)>
]>
<message >
 <header>
  <date>
   <day>12</day>
  <month type="character">MAY</month>
   <year>2006</year>
  </date>
  <from> luliuyan </from>
  <to>    luliuyan's wife</to>
  <to relationship="very close">Your family</to>
  <to relationship="not so close">Your friends</to>
```

```
  <subject> Merry Christmas</subject>
  <banner></banner>
 </header>
 <body>
  <paragraph size="1 line">
  Best wishes for Christmas
  </paragraph>
  <paragraph size="2 line">
  I love you!!
  </paragraph>
 </body>
 <footer>
  <day>Christmas Day</day>
  <from>The best for us</from>
 </footer>
</message>
```

从上面的示例可以得出以下几点结论。

- 第 1 层 message：包含 header、body、signature 3 个元素，或者 footer（其中一个）。
- 第 2 层 header：又包含 date、from、to（“+” 表示 to 元素出现次数为一次到多次）、subject、banner（“?”表示 banner 元素出现次数为 0 到多次）。body 有子元素 paragraph（"*"表示 paragraph 元素可以不出现，或出现一次或多次），而且 paragraph 有 #PCDATA 限制，表示不能包含其他元素，且只能是字符数据。另外，paragraph 含有属性 size，且 size 属性的类型是 CDATA，表示是字符数据，#REQUIRED 表示使用 paragraph 元素时必须使用 size 属性。signature 元素也为字符类型。footer 元素为任意类型 ANY（本例中是包含其他子元素的类型）。
- 第 3 层 date：包含 day、month（有 type 属性）、year（有 format 属性）3 个子元素；from 元素和 to 元素有 relationship 属性。

2．XML 数据底层结构的 Schema

Schema 的优点其实就是 DTD 的缺点：

- DTD 是使用与 XML 不同的语法写的，而 Schema 使用的是一种类似 XML 的语言。
- DTD 中所有声明都是全局的，而 Schema 既有全局声明，又有局部声明。
- Schema 最大的优点就是支持数据类型，而且支持自定义的数据类型。

另外，Schema 模式是个文本文件，独立于 XML 文档，以.xsd 为文件扩展名。

例 2.29　Schema 示例。

studInfo.xsd 文件的内容如下：

```
<?xml version="1.0" encoding="GB2312"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:annotation>
  <xsd:documentation>
  这是一个关于学生个人信息的 XML 文档
```



```
  </xsd:documentation>
 </xsd:annotation>
 <xsd:element name="studInfo" type="studInfoType"/>
 <xsd:complexType name="studInfoType">
  <xsd:sequence>

  <xsd:element name="studID" type="xsd:string"/>
  <xsd:element name="studName" type="xsd:string"/>
  <xsd:element name="studAge"   type="xsd:integer"/>
  <xsd:element name="studAddress" type="addressType"/>
  </xsd:sequence>
 </xsd:complexType>
   <xsd:complexType name="addressType">
 <xsd:sequence>
  <xsd:element name="street" type="xsd:string"/>
  <xsd:element name="city" type="xsd:string"/>
  <xsd:element name="state"   type="xsd:string"/>
  <xsd:element name="phone" type="xsd:string" use="optional"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:schema>
```

studInfo.xml 文件的内容如下：

```
<?xml version="1.0" encoding="GB2312"?>
<studInfo studID="040310125">
 <studName>luliuyan</studName>
 <studAge>22</studAge>
 <studAddress>
  <street>168</street>
  <city>南京</city>
  <state>中国</state>
  <phone>02552113125</phone>
 </studAddress>
</studInfo>
```

3．Schema 和 DTD 的区别

XML Schema 和 DTD 都用于文档验证，但两者有一定的区别。其中 DTD 不遵守 XML 语法，不可扩展，不支持命名空间的应用，没有提供强大的数据类型支持，只能表示很简单的数据类型；Schema 完全基于 XML 语法，能用处理 XML 文档的工具处理，大大扩充了数据类型，支持原型和属性组，有开放性，可以多个 Schema 运用于一个 XML 文档。因此 Schema 更为精确和灵活。

2.3.4 XML 文件的设计

XML 文档的生成步骤如下：

（1）确定各种元素。

（2）根据各元素之间的关系生成结构树。

（3）根据结构树生成 XML Schema。

（4）生成 XML 文档。

命名空间是 W3C 推荐标准提供的一种统一命名 XML 文档中的元素和属性的机制。使用命名空间可以明确标识和组合 XML 文档中来自不同标记词汇表的元素和属性，避免了名称之间冲突而带来的问题。声明命名空间通常使用一个简短的代号来代替 URI，这个简短的代号称为命名空间前缀，由编写 XML 文档的人员自由决定。前缀只能包含 XML 标准中规定允许用作元素和属性名的字符组成，其中包括英文字母和所有收录在 Unicode 中的汉字。命名空间声明的一般形式如图 2-1 所示。

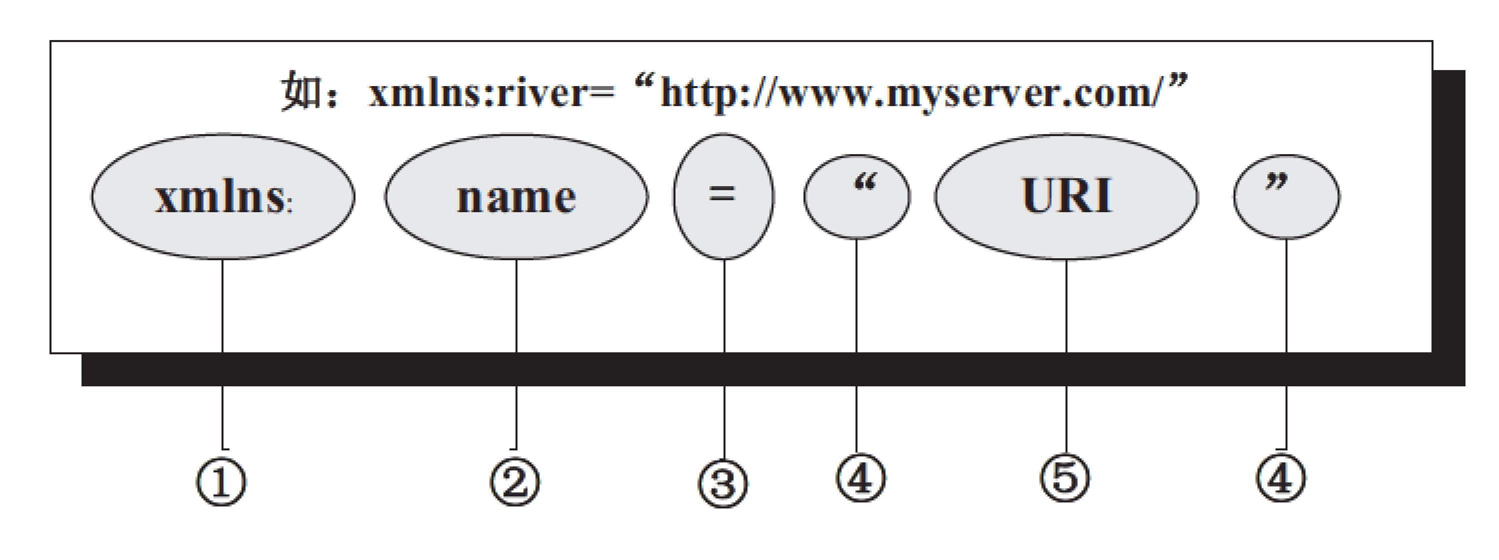


图 2-1　命名空间声明

第①部分是一个关键字“xmlns:”，第②部分是命名空间的前缀，第③部分是一个等号，第④部分是双引号（即把第⑤部分的名空间标识 URI 包括起来），第⑤部分是名空间标识 URI。需要注意的是，命名空间的前缀不能称为 xml，因为在 XML 中这个字符串是保留作特殊用途的，如 xml:space。另外，还可以隐式声明命名空间，即省略掉冒号和命名空间前缀，如图 2-2 所示。

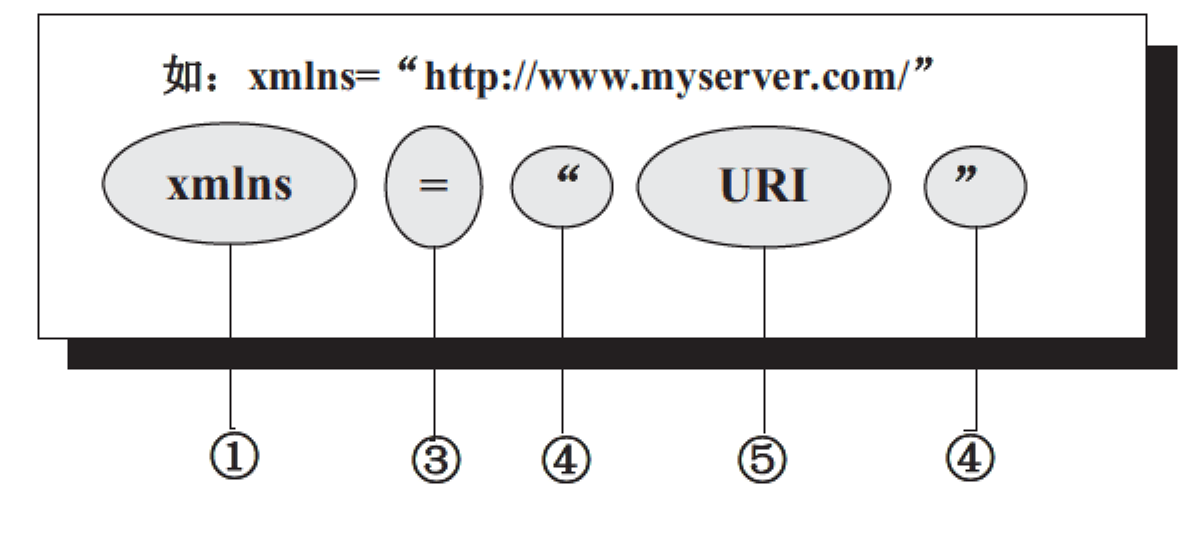


图 2-2　命名空间声明

在 XML 中，命名空间的使用涉及“范畴”的概念，范畴即命名空间的覆盖范围，它指的是哪些元素和属性在该命名空间中，哪些不在。命名空间既可以限定整个 XML 文档，也可以只针对 XML 文档中的一部分，另外，在 XML Schema 中也使用命名空间。



2.3.5　XML 与 Java

许多网络开发人员认为，XML 和 Java 的相配合是最合适的选择，是因为 XML 和 Java 能够相互补充而构成一个完整的开发平台。XML 能够提供平台无关的数据，而 Java 提供平台无关的软件处理。在 1997 年 6 月，Sun 微系统公司的 Jon Bosak 在他的"XML, Java, and the Future of the Web"论文中提到："XML 能够让 Java 实现更多的事情。"也就是说，在一些特殊的应用领域中，XML 为 Java 提供了供处理的数据。XML 和 Java 对 Internet 都是很友好的，它们内部都对 Internet 进行了特别的优化。XML 在设计之初，就是向着一种能够在 Internet 上自由传播的、优化的、有弹性的、易于阅读的格式这样一个目标设计的。Java 中更是内建了与许多网络相关的类库，能够很好地支持 sockets、HTTP、HTML 等网络协议，它们都支持 Unicode，因而开发出来的应用也很容易进行国际化。Java 为程序员提供了表达复杂数据结构以及面向对象的建模方法，而 XML 在这方面也是一个比较理想的工具。虽然 XML 出现不是很久，但现在支持 XML 的工具越来越多，从编写 XML 到 XML 的 Paser 解析器一应俱全，这也为 XML 的发展奠定了很好的基础。

2.3.6　XML 与.NET

Internet 的应用正在不断地扩大，能用某种方法访问远程站点提供的计算能力并利用它的服务，而不仅仅是利用它的发布能力，就是.NET 将要提供的精华。

.NET 允许在服务的层次上，而不是在发布的层次上来共享信息并交互。Microsoft 的支持.NET 的产品和许多来自第三方的组件、内容和功能都拥有.NET 的内部结构，但自然地会提出一个问题——是什么使.NET 工作呢？这些都是从 XML 开始的。要想理解.NET，就需要理解 XML。像我们在交流中所说和写的语言一样，XML（EXtensible Markup Language，可扩展标记语言）是.NET 的基础，是.NET 的灵魂，是所有.NET 现在和将来的基础。数据库将通过 XML 中的记录集来读写，Web 浏览器将接受 XML 并将其和伴随它的样式表一起显示，Visual Studio 甚至会产生 XML 代码！不理解 XML 和与之相关的技术，就不能支持.NET 的资源交流，无论是站点还是人！对于今天的 Web 服务器而言，XML 差不多无所不在。几乎所有的计算平台都能分析 XML，因而也就能获得 XML 文档中的内容，Windows 能，Linux 能，当然 MVS 和 VMS 也能，甚至蜂窝式移动电话也能。

2.3.7　XML 应用实例

XML 在不同领域有着广泛的应用，例如，在科技领域的 MathML、无线通信应用的 WML 和在网络图像方面的 SVG 等，这里侧重讨论 XML 在 Web 上的应用。XML 在 Web 上应用主要是利用其强大的数据操作能力。一般用 XML 配合 JavaScript 和 ASP 等服务器端

程序，可以实现网络上几乎所有的应用需求，这是一个简单的书目数据检索功能。通过单击“上一个”、“下一个”按钮可以浏览每本书的有关信息。下面分析它的设计过程。

1. 定义新标识

根据实际的书目数据，首先新建一个名为<CD>的标识；然后建立它相关的数据标识，即 CD 名称<Title>、作者<Artist>、出版年代<Year>、国家<Country>、出版社<Company>和价格<Price>；最后还要建立一个名为目录<CATALOG>的标识。为什么要再建立一个<CATALOG>标识呢？因为在 XML 文档中规定，必须且只能有一个根元素（标识），我们有多个 CD 数据，这些数据是并列的关系，所以需要为这些并列的元素建立一个根元素。以上元素的定义和关系都完全符合 XML 标准，不需要特别的 DTD 文件来定义，所以可以省略 DTD 定义。如果想使用 DTD 来定义，以上过程可以用下列代码表示：

```
<!ELEMENT CATALOG (CD)*>
<!ELEMENT CD (Title,Artist,Year,Country,Company,Price)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Artist (#PCDATA)>
<!ELEMENT Year (#PCDATA)>
<!ELEMENT Country (#PCDATA)>
<!ELEMENT Company (#PCDATA)>
<!ELEMENT Price (#PCDATA)>
```

上述代码表示：元素 CATALOG 包含多个 CD 子元素，而子元素 CD 又依次包含 Title、Artist、Year、Country、Company、Price 6 个子元素，它们的内容都定义为文本。

2. 建立 XML 文档

建立 XML 文档的代码如下：

```
<?xml version="1.0"?><CATALOG>
<CD>
<TITLE>Empire Burlesque</TITLE>
<ARTIST>Bob Dylan</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Columbia</COMPANY>
<PRICE>10.90</PRICE>
<YEAR>1985</YEAR>
</CD>
<CD>
<TITLE>Hide your heart</TITLE>
<ARTIST>Bonnie Tylor</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>CBS Records</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1988</YEAR>
```



```
</CD>
<CD>
<TITLE>Greatest Hits</TITLE>
<ARTIST>Dolly Parton</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>RCA</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1982</YEAR>
</CD>
<CD>
<TITLE>Still got the blues</TITLE>
<ARTIST>Gary More</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>Virgin redords</COMPANY>
<PRICE>10.20</PRICE>
<YEAR>1990</YEAR>
</CD>
<CD>
<TITLE>Eros</TITLE>
<ARTIST>Eros Ramazzotti</ARTIST>
<COUNTRY>EU</COUNTRY>
<COMPANY>BMG</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1997</YEAR>
</CD>
</CATALOG>
```

上面代码首先用“<?xml version="1.0"?>”声明语句表明这是一个 XML 文档，它的格式遵守 XML 1.0 标准规范。文档内容的代码如下：

```
<CATALOG>
<CD>
...
</CD>
<CD>
...
</CD>
</CATALOG>
```

此时，一共定义了 5 组数据。将上面的代码存为 cd.xml 文件，以备调用。

3. 建立相应的 HTML 文件

（1）导入 XML 数据

IE 是通过在 HTML 中的 object 来支持插入 XML 的，并通过 XMLDocument.load()方法来导入数据。代码如下：

```
<object WIDTH="0" HEIGHT="0"
CLASSID="clsid:550dda30-0541-11d2-9ca9-0060b0ec3d39" ID="xmldso">
</object>
```

定义一个 object，ID 名为 xmldso，然后在 head 区引入 XML 数据，代码如下：

```
<script for="window" event="onload">
xmldso.XMLDocument.load("cd.xml");
</script>
```

（2）捆绑数据

用<SPAN>标识将 XML 数据绑定在表格中。其中 ID、DATASRC、DTATFLD 都是<SPAN>的属性，代码如下：

```
<table>
<tr><td>Title:</td><td><SPAN ID="title" DATASRC=#xmldso DATAFLD="TITLE"></SPAN></td></tr>
<tr><td>Artist:</td><td><SPAN ID="artist" DATASRC=#xmldso DATAFLD="ARTIST"></SPAN></td></tr>
<tr><td>Year:</td><td><SPAN ID="year" DATASRC=#xmldso DATAFLD="YEAR"></SPAN></td></tr>
<tr><td>Country:</td><td><SPAN ID="country" DATASRC=#xmldso DATAFLD="COUNTRY"></SPAN>
</td></tr>
<tr><td>Company:</td><td><SPAN ID="company" DATASRC=#xmldso DATAFLD="COMPANY">
</SPAN> </td></tr>
<tr><td>Price:</td><td><SPAN ID="price" DATASRC=#xmldso DATAFLD="PRICE"></SPAN></td></tr>
</table>
```

4. 为数据提供浏览按钮

代码如下：

```
<INPUT TYPE=button VALUE="上一张 CD" ONCLICK="moveprevious()">
<INPUT TYPE=button VALUE="下一张 CD" ONCLICK="movenext()">
```

利用 movenext()和 moveprevious()完成两个鼠标的点击功能，在 head 区加入如下代码：

```
<script language="JavaScript">
function movenext()
{
if (xmldso.recordset.absoluteposition < xmldso.recordset.recordcount)
{
xmldso.recordset.movenext();
}
}
function moveprevious()
{
if (xmldso.recordset.absoluteposition > 1)
{
xmldso.recordset.moveprevious();}
</script>
```



下面是 HTML 文件的全部源代码：

```
<html>
<head>
<script for="window" event="onload">
xmldso.XMLDocument.load("cd.xml");
</script>
<script language="JavaScript">
function movenext()
{
if (xmldso.recordset.absoluteposition < xmldso.recordset.recordcount)
{
xmldso.recordset.movenext();}
}function moveprevious()
{
if (xmldso.recordset.absoluteposition > 1)
{
xmldso.recordset.moveprevious();
}
}
</script>
<title>CD Navigate</title>
</head>
<body>
<p>
<object WIDTH="0" HEIGHT="0"
CLASSID="clsid:550dda30-0541-11d2-9ca9-0060b0ec3d39" ID="xmldso">
</object>
<table>
<tr><td>Title:</td><td><SPAN ID="title" DATASRC=#xmldso DATAFLD="TITLE"></SPAN></td></tr>
<tr><td>Artist:</td><td><SPAN ID="artist" DATASRC=#xmldso DATAFLD="ARTIST"></SPAN></td></tr>
<tr><td>Year:</td><td><SPAN ID="year" DATASRC=#xmldso DATAFLD="YEAR"></SPAN></td></tr>
<tr><td>Country:</td><td><SPAN ID="country" DATASRC=#xmldso DATAFLD="COUNTRY">
</SPAN></td></tr>
<tr><td>Company:</td><td><SPAN ID="company" DATASRC=#xmldso DATAFLD="COMPANY">
</SPAN></td></tr>
<tr><td>Price:</td><td><SPAN ID="price" DATASRC=#xmldso DATAFLD="PRICE"></SPAN></td></tr>
</table>
<p>
<INPUT TYPE=button VALUE="上一张 CD" ONCLICK="moveprevious()">
<INPUT TYPE=button VALUE="下一张 CD" ONCLICK="movenext()">
</p>
</body>
</html>
```

将以上代码存为 cd.htm 文件，与第 2 步的 cd.xml 文件放在一起。打开 cd.htm 文件，就可实现该功能。

2.4 小　　结

本章首先介绍了标记语言的主要发展历程，然后重点介绍了超文本标记 HTML 的基础知识，包括超链接、列表、表格、表单、框架、图像、文本格式以及网页设计等，并对 XML 的文档类型、底层结构以及与 Java、.NET 的关系进行了详细的介绍，最后列举了一个 XML 应用实例。通过本章学习，读者可以全面了解 HTML 和 XML 概念，为本书后面的学习打下坚实的基础。

2.5 思 考 题

1．什么是 HTML？它与 XML 有什么本质的不同？
2．在 XML 中，文档类型定义 DTD 和文档结构的目的是什么？
3．标记一个图片的超链接应如何表示？
4．超链接标记有哪些属性？每种属性的含义是什么？
5．什么是 XML 的名称空间？

第 3 章　网页与网站设计基础

网页是存储在 Web 服务器上的一个个 HTML、JSP、ASP、PHP 等各种类型的文档，并且能够在浏览器中以多媒体的形式呈现出来。网站是由大量的网页组成的多媒体资源。那么网页设计的原则是什么？常见的制作工具有哪些？网站如何规划管理呢？

本章主要介绍 3 个方面的内容：网页的基础知识，包括什么是网页、网页的基本元素、网页设计原则；常见的网页制作工具，如网页制作软件、图形图像处理软件、动画设计软件等；网站规划设计基础，包括网站设计流程、确定网站的类型以及确定网站栏目和板块等。

3.1　网页的基础知识

3.1.1　什么是网页

网页也称为 Web 页，是 WWW 的基本文档，它由文字、图片、动画和声音等多种媒体信息以及超链接组成，是用 HTML（超文本标识语言）或者其他语言（如 JavaScript、VBScript、ASP、PHP、XML 等）编写而成的。网页通过链接实现与其他网页或网站的关联和跳转。如图 3-1 所示为清华大学网站的主页。



图 3-1　清华大学网站主页

网页是网站的基本信息单位，一个网站通常由许多网页组成，而众多网页中处于第一

页（首页）的页面被称为主页，从主页可以链接到其他的网页。通常所说的 Home Page 是指首页，即进入网站其他页面的“入口”。一个网站可以有多个网页，但只能有一个首页。从文件角度讲，网页是一种由 HTML 语言编写的文本文件，存放在世界某个角落的某一台计算机中，而这个计算机必须是与互联网相连接的，所以网页文件又称为 HTML 文件，其扩展名为.html 或.htm，网页的 HTML 源代码如图 3-2 所示。网页作为一种文本文件，可以用任意文本编辑器编辑，如可使用 Windows 系统中的“记事本”程序。

```
index[1] - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
<html>
<head>
<title>欢迎光临清华大学</title>
<link href="../cic_jsp/qhdwzy/css/main1.css" rel="stylesheet" type="text/css">
<style type="text/css">
<!--
.STYLE1 {
        color: #990066;
        font-size: 14px;
}
body {
        background-color: #FFFFFF;
}
.menu { width:1000px; height:66px; background:url(/cic_jsp/qhdwzy/index_images/menu_bg37.gif) no-
repeat; font-size:12px; font-family:"宋体", "幼圆"; margin:0; }
.menu ul { height:50px; margin-top:5px; margin-left:105px; *margin-left:140px!important; _margin-
left:140px; }
.menu ul li { list-style:none; float:left; }
.menu ul .btn01 { width:145px; height:50px; line-height:22px; background:url
(/cic_jsp/qhdwzy/index_images/btn_bg01_b.gif) no-repeat; text-align:center; padding-right:25px; }
.menu ul .btn02 { width:145px; height:50px; line-height:22px; background:url
(/cic_jsp/qhdwzy/index_images/btn_bg.gif) no-repeat; text-align:center; padding-right:20px; }
.menu ul li a { width:145px; height:50px; display:block; color:#fff; text-decoration:none; }
.menu ul li a:hover { color:#ff0; }
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"></head>
<body><center>
  <table width="1000" height="65" border="0" cellpadding="0" cellspacing="0">
    <tr align="left" valign="top">
      <td width="131"><img src="/cic_jsp/qhdwzy/index_images/index_01.gif" width="337" height="65"
alt=""></td>
      <td width="871"> </td>
    </tr>
  </table>
```

图 3-2　网页的 HTML 源代码

3.1.2　网页的基本元素

构成网页的主要元素有文字、图像和超链接，文字和图像是用户传播信息的主要形式，而超链接则是不同页面的纽带，3 个部分有机结合，就组成了完整的页面。另外，其他一些辅助页面元素的加入，也使得页面越来越丰富多彩。

1. 文字

文字是网页中最基本的元素，也是网页发布信息所用的最主要形式。由文字制作出的网页占用空间小，因此下载速度很快。另外，文字性网页还可以利用浏览器中“文件”菜单下的“另存为”功能将其下载，便于以后长期阅读，也可对其进行编辑、打印。但是没有编排点缀的纯文字网页，美观性很差，所以，网页上的文字可以设置不同的大小、字体、颜色和格式等，以使页面更生动。也可以采用一些动态技术，使文字具有各种各样的动态效果。

2. 图像

图像是网页必不可少的另一个重要元素，图像传递的信息比文字更直观，在网页中适当添加图像，还可以对网页起到装饰和美化的作用，能给用户带来更为强烈的视觉效果。

在页面上可以使用的图像格式主要有 GIF、JPG/JPEG、PNG、BMP 等，其中最常用的



是 GIF 和 JPG/JPEG，这两种格式具有跨平台的特性，可以在不同操作系统支持的浏览器上显示。

3. 超链接

超链接是建立在同一站点或不同站点中 Web 页之间的跳转关系，是 WWW 中最具特色的功能。超链接的设置，使得用户在浏览网页时可以随意转向其他有兴趣的页面，整个站点的网页都成为有机的整体，而且站点与站点之间也可以建立联系。

超链接的建立，是通过链接载体和链接目标的设置实现的。页面中的很多元素都可以作为链接载体，如文字、图像、图像热点区域和动画等，而链接目标则是任意的网络资源，如页面、图像、声音、程序、E-mail 甚至是页面中的某个位置——锚点。

4. 声音和动画

声音和动画也是网页上的重要表现形式，适当地使用这些多媒体元素，可以使网页更生动。声音在网页中主要应用于网页背景或者音乐站点。网页中支持的声音格式有 MID、MP3、WAV 和 RM 等。动画文件的格式有 AVI 视频、Java Applet 动画等，最常用的动画文件是 Flash 动画，但是 Flash 动画需要安装插件才可以播放。

5. 表单

表单可以为浏览器和服务器之间提供信息交流，通常用于填写申请或提交信息的交互页面，如用户注册页面、电子邮件等的申请页面和论坛、留言簿等进行数据提交的页面都是通过表单实现的，如图 3-3 所示。

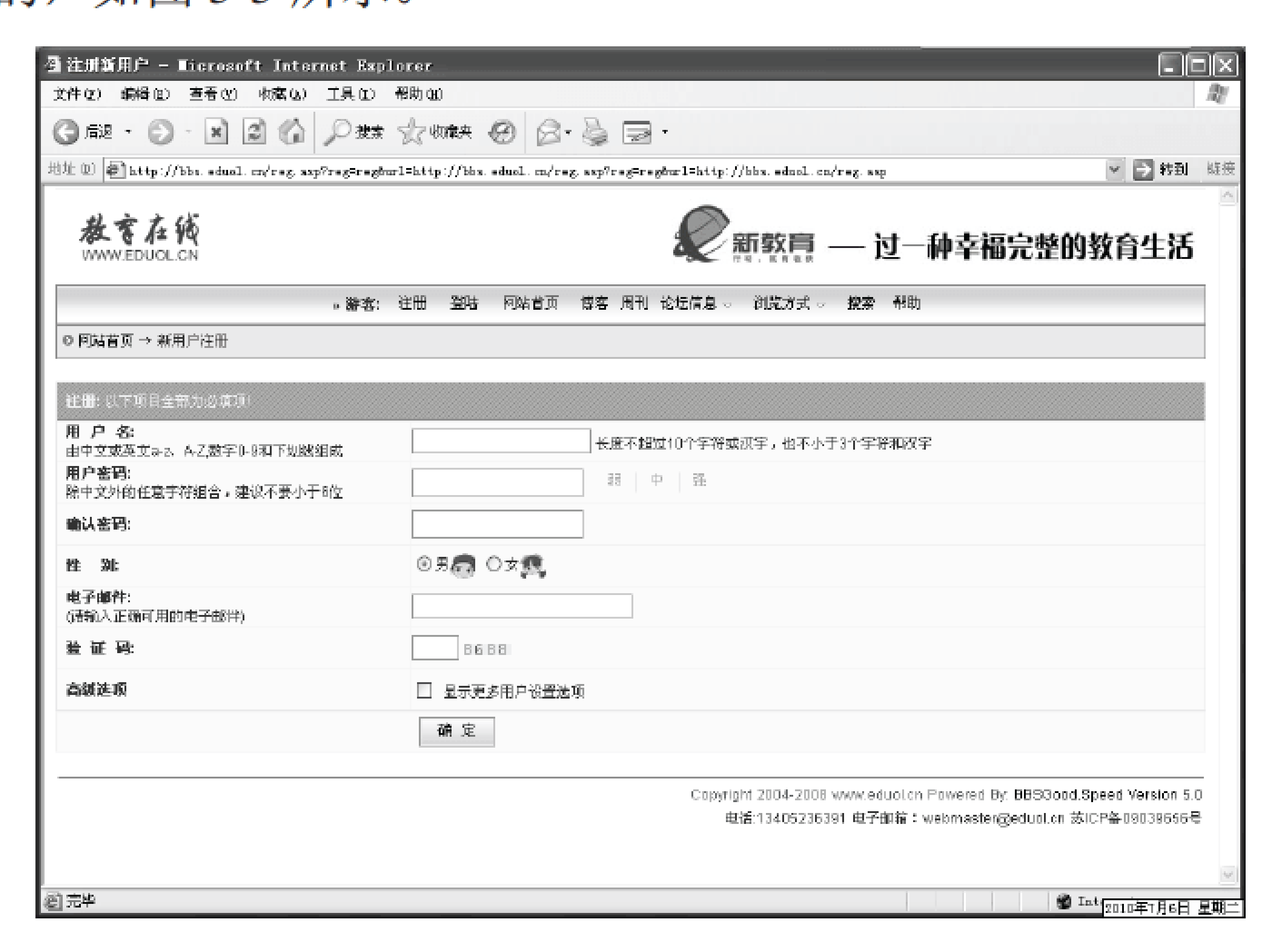


图 3-3　用户注册表单页面

3.1.3　网页设计原则

网页的美观与实用，关系到网站的整体浏览效果，良好的网页需要通过网页设计与网

页制作两个步骤完成，因此，网页设计是一个感性思考与理性分析相结合的复杂过程，它的设计取决于整个网站的风格，它的实现依赖于网页制作。

网页设计依赖于设计者对整个网站的理解及设计制作水平，而且还与设计者自身的审美观以及对页面的把握有关。在设计时要遵循以下网页设计原则。

1．内容与形式要统一

网页的内容与形式是相互依赖的，设计作品的整体效果是至关重要的。网页提供给浏览者最主要的是内容，因为用户访问网站的最终目的是获取自己想要的信息，但是单纯的内容又使页面显得枯燥无味。在设计时，页面的内容与形式应相互对应、相互衬托，内容丰富且外观漂亮的页面才能使用户喜欢。

2．页面之间的超链接要有效

页面之间的跳转是通过超链接实现的，超链接的实现效果严重影响整个网站的效果，也会影响访问者的兴趣。如果访问者在单击超链接之后，弹出的页面无法显示，或正在建设中，那么访问者是不可能再有心情去继续欣赏这个网站的。为了保证网站的整体效果，在制作时要将设置的各个超链接真实完整地实现。

3．导航结构要清晰

网页中的导航为访问者提供了浏览整个网站的“路标”。合理的“路标”能够让访问者快速、准确地找到自己所期望的内容。所有的超链接要清晰无误地标识出来，所有导航性质的设置，如图像按钮、链接文字等都要有清晰的标识，当然，也要兼顾美观的视觉效果。

4．访问速度要流畅

排除服务器的因素，决定网页访问速度的主要是网页的大小以及网页中各种元素所占空间的大小。在访问者打开想要浏览的网页时，如果网页文件较大，页面显示的速度必然会很慢，在等待片刻之后，页面还没有完整打开，那么，访问者对该网页的兴趣就会大大降低，甚至会直接关闭。所以，通常情况下，要求页面代码、图片及页面中的多媒体元素都要优化处理，可采用的措施有选择合适大小的图片添加在网页中、Flash 动画也要尽可能短、尽量少地使用背景音乐等。

3.2　常用的网页制作工具

3.2.1　网页制作软件

（1）FrontPage

FrontPage 是由 Microsoft 公司推出的 Web 网页制作工具。FrontPage 使网页制作者能够



更加方便、快捷地创建和发布网页，具有直观的网页制作和管理方法，简化了大量工作。FrontPage 界面与 Word、PowerPoint 等软件的界面极为相似，为使用者带来了极大的方便，Microsoft 公司将 FrontPage 封装入 Office 中，使其成为 Office 家族的一员，功能更为强大。

FrontPage 的这种“所见即所得”的操作方式能够让操作者在没有 HTML 语言基础的情况下很快掌握相关操作。

但 FrontPage 也有不足之处，如浏览器兼容性不好，做出来的网页用 Netscape 往往不能正常显示，另外，生成的垃圾代码多，也会自动修改代码，导致在某些情况下极为不便，还有就是对 DHTML 的支持不好。但随着这几年的版本升级，FrontPage 经历了 FrontPage 98、FrontPage 2000、FrontPage 2002、FrontPage XP 到现在的 FrontPage 2003 版本，不足之处得到了不断的完善。所以说，FrontPage 是比较好的入门级网页编辑工具。

（2）Dreamweaver

Dreamweaver 是美国 Macromedia 公司（后被 Adobe 公司收购）开发的集网页制作和管理网站于一身的所见即所得网页编辑器，它是第一套针对专业网页设计师特别发展的视觉化网页开发工具，利用它可以轻而易举地制作出跨越平台限制和浏览器限制的充满动感的网页。

Dreamweaver 包括可视化编辑、HTML 代码编辑的软件包，并支持 ActiveX、JavaScript、Java、Flash、ShockWave 等特性，而且它还能通过拖拽从头到尾地制作动态的 HTML 动画，并支持动态 DHTML（Dynamic HTML）的设计，可以轻而易举地做出很多眩目的页面特效。Dreamweaver 是建立 Web 站点和应用程序的专业工具，其网页动态效果与网页排版功能都比一般的软件好用。

Dreamweaver 支持动态 HTML，并采用了 Roundtrip HTML 技术，从而奠定了在网页高级设计功能方面的领先地位。在进行网页设计的过程中，动态 HTML 技术能够让用户轻松设计复杂的交互式网页，产生动态效果；而 Roundtrip HTML 技术则可以真正支持 HTML 源编辑模式，不会产生冗余代码，使网页渲染速度加快。因此，Dreamweaver 是一种可以满足多层次需求、功能强大的可视化专业级网页设计及制作工具。

Dreamweaver 的版本从最初的 Dreamweaver 3、Dreamweaver 4、Dreamweaver MX、Dreamweaver MX2004、Dreamweaver 8，发展到现在的 Adobe Dreamweaver CS3（MacroMedia 被 Adobe 收购后推出的第一个版本）、Adobe Dreamweaver CS4。2010 年 4 月 12 日备受关注的 Adobe 新一代产品 Creative Suite 5（CS5）正式发布了，其中包括了新的 Dreamweaver CS5。

3.2.2　图形图像处理软件

图形图像处理软件是对数字图片进行修复、合成、美化等各种处理的软件的总称。图形图像处理软件被广泛应用于广告制作、平面设计和影视后期制作等领域。下面介绍常见的图像图形处理软件。

（1）Photoshop

Photoshop 是 Adobe 公司推出的一款功能强大的图像处理软件，它是目前公认的 PC 机上最好的通用平面美术设计软件，它界面简洁友好、可操作性强、功能完善、性能稳定，

所以备受广大图形设计爱好者和专业图像设计人员的青睐，被广泛地应用在图像处理、绘画、多媒体界面设计和网页设计等领域。Adobe Photoshop 的版本从最初的 3.0、4.0、5.0、6.0、7.0，发展到后来推出的 Adobe Photoshop CS、CS2、CS3、CS4，现在最新版为 Adobe Photoshop CS5。

（2）Fireworks

Adobe Fireworks 是 Adobe 推出的一款网页作图软件，软件可以加速 Web 设计与开发，是一款创建与优化 Web 图像和快速构建网站与 Web 界面原型的理想工具。Fireworks CS4 不仅具备编辑矢量图形与位图图像的灵活性，还提供了一个预先构建资源的公用库，目前，广泛使用的 Adobe Fireworks CS4 可与 Adobe Photoshop CS4、Adobe Illustrator CS4、Adobe Dreamweaver CS4 和 Adobe Flash CS4 软件集成。在 Fireworks 中将设计迅速转变为模型，或利用来自 Illustrator、Photoshop 和 Flash 的其他资源，然后直接置入 Dreamweaver CS4 中轻松地进行开发与部署。

作为一款为网络设计而开发的图像处理软件，Fireworks 还能够自动切割图像、生成光标动态感应的 JavaScript 程序等，而且 Fireworks 具有强大的动画功能和一个相当完美的网络图像生成器。

（3）AutoCAD

AutoCAD 是由美国 Autodesk 公司于 20 世纪 80 年代初为微机上应用 CAD 技术而开发的绘图程序软件包，经过不断地完善，现已经成为国际上广为流行的绘图工具。

AutoCAD 具有良好的用户界面，通过交互菜单或命令行方式便可以进行各种操作。它的多文档设计环境，让非计算机专业人员也能很快学会使用。在不断实践的过程中更好地掌握它的各种应用和开发技巧，从而不断提高工作效率。

AutoCAD 具有广泛的适应性，它可以在各种操作系统支持的微型计算机和工作站上运行，并支持分辨率在 320×200 到 2048×1024 之间的各种图形显示设备 40 多种，以及数字仪和鼠标器 30 多种、绘图仪和打印机数十种，这就为 AutoCAD 的普及创造了条件。现在最新的版本为 AutoCAD 2011。

（4）CorelDRAW

CorelDRAW Graphics Suite 是一款由世界顶尖软件公司之一的加拿大的 Corel 公司开发的图形图像软件。其非凡的设计能力广泛地应用于商标设计、标志制作、模型绘制、插图描画、排版及分色输出等诸多领域。

CorelDRAW 界面设计友好、空间广阔、操作精微细致，它提供给设计者一整套的绘图工具，包括圆形、矩形、多边形、方格和螺旋线等，并配合塑形工具，可以作出更多的变化，如圆角、矩形、弧、扇形、星形等。同时也提供了特殊笔刷，如压力笔、书写笔和喷洒器等，以便充分地利用电脑处理信息量大、随机控制能力高的特点。目前常用的是 CorelDRAW X4 版本。

3.2.3 动画设计软件

常用的动画设计软件有 Flash、3D Studio Max 和 Maya 等。



（1）Flash

Flash 是美国 Macromedia 公司所设计的一种二维矢量动画软件（现 Adobe 公司产品），用于设计和编辑 Flash 文档，最新版本为 Adobe Flash CS5。Flash 通常也指 Macromedia Flash Player（现 Adobe Flash Player），用于播放 Flash 文档。

Flash 是一种交互式动画设计工具，用它可以将音乐、声效、动画以及富有新意的界面融合在一起，以制作出高品质的网页动态效果。它主要应用于网页设计和多媒体创作等领域，功能十分强大和独特，已成为交互式矢量动画的标准，在网上非常流行。Flash 广泛应用于网页动画制作、教学动画演示、网上购物和在线游戏等的制作中。

（2）3D Studio Max

3D Studio Max 常简称为 3ds max 或 MAX，是 Autodesk 公司开发的基于 PC 系统的三维动画渲染和制作软件，其前身是基于 DOS 操作系统的 3D Studio 系列软件，最新版本是 2010。在 Windows NT 出现以前，工业级的 CG 制作被 SGI 图形工作站所垄断。3D Studio Max+Windows NT 组合的出现降低了 CG 制作的门槛，首先开始运用在电脑游戏中的动画制作，然后更进一步开始参与影视片的特效制作，如 X 战警 II、最后的武士等。

在应用范围方面，3D Studio Max 广泛应用于广告、影视、工业设计、建筑设计、多媒体制作、游戏、辅助教学以及工程可视化等领域。拥有强大功能的 3ds max 被广泛地应用于电视及娱乐业中，如片头动画和视频游戏的制作，在影视特效方面也有一定的应用。而在国内发展的相对比较成熟的建筑效果图和建筑动画制作中，3ds max 的使用率更是占据了绝对的优势。

（3）Maya

Maya 是美国 Autodesk 公司出品的世界顶级的三维动画软件，应用对象是专业的影视广告、角色动画和电影特技等。Maya 功能完善、工作灵活、易学易用、制作效率极高、渲染真实感极强，是电影级别的高端制作软件。Maya 售价高昂，声名显赫，是制作者梦寐以求的制作工具，掌握了 Maya，会极大地提高制作效率和品质，调节出仿真的角色动画，渲染出电影一般的真实效果，向世界顶级动画师迈进。

Maya 集成了 Alias/Wavefront 最先进的动画及数字效果技术，不仅包括一般三维和视觉效果制作的功能，而且还与最先进的建模、数字化布料模拟、毛发渲染、运动匹配技术相结合。Maya 可在 Windows NI 与 SGI IRIX 操作系统上运行。在目前市场上用来进行数字和三维制作的工具中，Maya 是首选解决方案。

3.2.4　动态网页制作技术

实现动态网页后，可以在网站中灵活使用数据库技术，为用户提供交互式的网页访问方式。制作动态网页一般以静态网页为基础，然后添加一些实现特定功能的服务器端脚本，一般要与数据库建立连接，通过数据库访问，实现相关功能。与动态网页相关的服务器技术有 CGI、ASP、PHP 及 JSP 等。

1. CGI

CGI（Common Gateway Interface，公共网关接口）是外部程序与 Web 服务器之间的标

准编程接口。可以使用不同的语言编写出 CGI 程序，然后放在 Web 服务器上，用户向 Web 服务器发出浏览请求时，会触发 CGI 程序的运行，Web 服务器将运行结果发给客户端浏览器。这种技术较浪费服务器资源，且效率较低。

2．ASP

ASP 指 Active Server Pages（动态服务器页面），是运行于 IIS 之中的程序，是一项微软公司的技术，是一种使嵌入网页中的脚本可由因特网服务器执行的服务器端脚本技术。

目前常用的是 ASP.NET 技术。ASP.NET 不仅是 ASP 的下一个版本，而且是一种建立在通用语言上的程序构架，能被用于一台 Web 服务器来建立强大的 Web 应用程序。ASP.NET 提供许多比现在的 Web 开发模式强大的优势，它是把基于通用语言的程序在服务器 IIS 上运行。不像以前的 ASP 即时解释程序，而是将程序在服务器端首次运行时进行编译，这样的执行效果当然比一条一条地解释强很多。但是 ASP.NET 也有一个特点，就是每修改一次程序（即代码类），必须重新编译一次，修改几次就必须重新编译几次，执行效果也会有所降低。

ASP.NET 构架可以用 Microsoft 公司的产品 Visual Studio.NET 开发环境进行开发。ASP.NET 一般分为两种开发语言，即 VB.NET 和 C#，C#相对比较常用。

3．PHP

PHP（Hypertext Preprocessor，超级文本预处理语言）是一种 HTML 内嵌式的语言，是一种在服务器端执行的嵌入 HTML 文档的脚本语言，语言的风格类似于 C 语言，被广泛运用。

PHP 独特的语法混合了 C、Java、Perl 以及 PHP 自创新的语法，它可以比 CGI 或者 Perl 更快速地执行动态网页。用 PHP 做出的动态页面与其他的编程语言相比，PHP 是将程序嵌入到 HTML 文档中去执行，执行效率比完全生成 HTML 标记的 CGI 要高许多；PHP 还可以执行编译后代码，编译可以加密并优化代码运行，使代码运行更快。PHP 具有非常强大的功能，所有 CGI 的功能 PHP 都能实现，而且支持几乎所有流行的数据库以及操作系统。

4．JSP

JSP（Java Server Pages）是由 Sun 公司推出的一种动态网页技术标准。JSP 技术有点类似于 ASP 技术，它是在传统的网页 HTML 文件（*.htm、*.html）中插入 Java 程序段（Scriptlet）和 JSP 标记（tag），从而形成 JSP 文件（*.jsp）。用 JSP 开发的 Web 应用是跨平台的，既能在 Linux 上运行，也能在其他操作系统上运行。

3.3　网站规划设计基础

多个网页按照一定的结构组织在一起就构成一个网站。每个网站都有一个主页，也是用户访问网站的入口。在建立网站之前，合理地规划设计站点是非常必要的工作。合理的



规划和设计，可以避免网站制作的盲目性，也有利于日后对网站的管理和维护。

3.3.1　网站设计流程

通常所见到的网站，虽然在规模、内容和功能上都有所不同，但是网站设计的基本过程都是一样的，通常应遵循的设计流程为：整体规划→网站设计→网页制作→测试与发布→网站维护。整个设计流程是一个循环的过程，如图 3-4 所示。

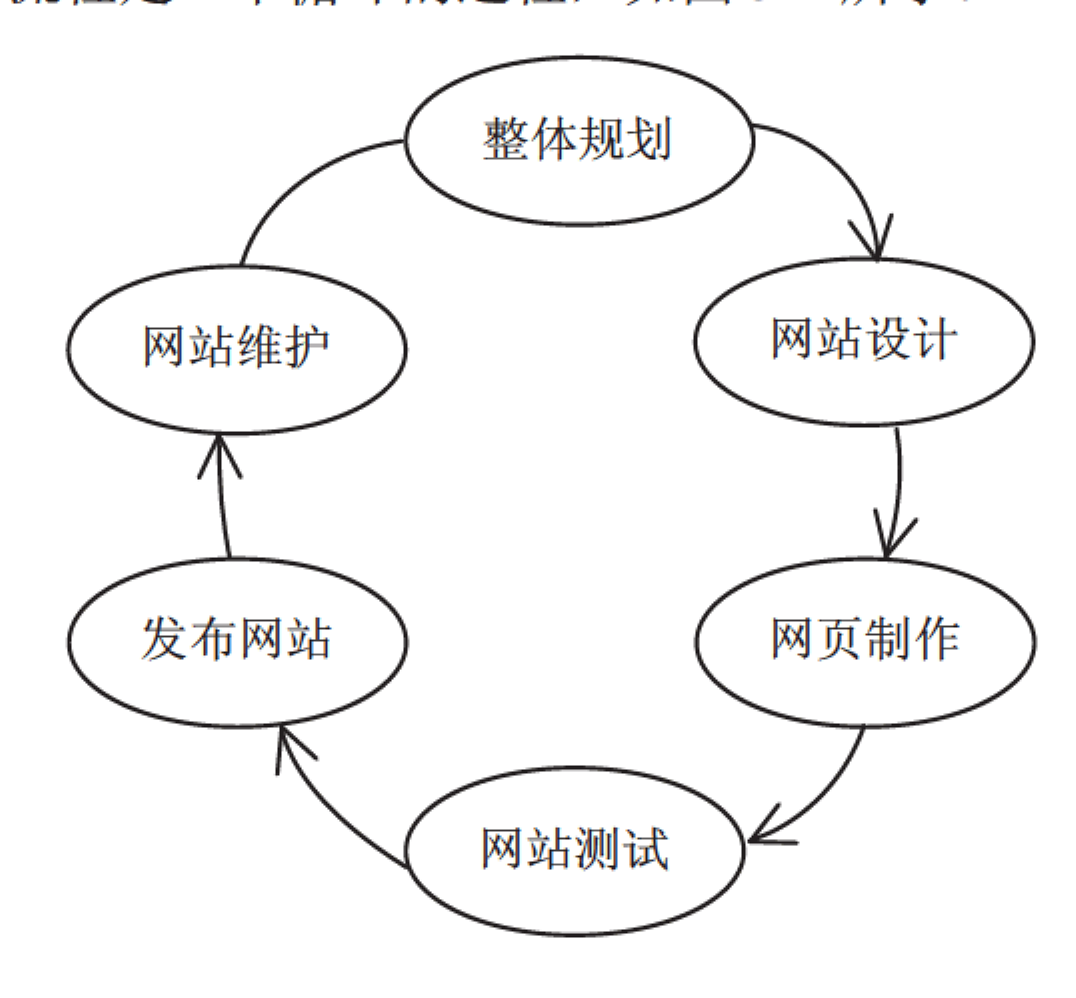


图 3-4　网站设计流程图

1. 整体规划

网站的整体规划在整个网站的设计流程中是尤为重要的。无论是大的门户网站还是只有几个页面的个人网站，都需要做好前期的策划工作，因此，在创建网站之前，应该合理地规划站点的主题、结构、内容、导航机制以及站点整体的风格等。

网站规划需要确定主题以及网站的风格，还要考虑技术因素（各种技术因素，包括网页下载速度、浏览器、分辨率和插件等）。

进行网站的整体规划也就是组织网站的内容和设计其结构。网页制作者在明确网页制作的目的及要包括的内容之后，接下来就应该对网站进行规划，以确保文件内容条理清楚、结构合理，这样不仅可以很好地体现设计者的意图，而且也将使网站可维护性与可扩展性增强。

2. 网站设计

网站规划好之后，就进入了设计阶段。利用规划阶段搜集的信息，在设计阶段需要对内容、导航系统等进行具体的设计。

设计的内容包括网站的内容结构，页面的布局，首页设计，导航系统，网页中的颜色、文字、图像、动画以及多媒体的使用等。

3. 网页制作

设计阶段完成后，进入网页制作阶段。

网页制作主要使用相应的软件，将规划设计的内容变成真实的网页。制作网页首先要根据设计结果制作出若干示范网页，然后通过 Dreamweaver 等软件制作具体页面，在网页中添加实际内容，包括文本、图像、声音、Flash 电影以及其他多媒体信息等。

4. 网站测试

当所有网站页面都制作完毕之后，就需要对所制作的网页进行审查和测试，测试内容包括功能性测试和完整性测试两个方面。

功能性测试就是要保证网页的可用性，达到最初的内容组织设计目标，实现所规定的功能，读者可方便快速地寻找到所需的内容。完整性测试就是保证页面内容显示正确、链接准确、无差错、无遗漏。

在测试网站时，除了对所有影响页面显示的细节因素进行一次测试外，页面中的超链接能否正常跳转也是一个值得重视的问题。

如果在测试过程中发现了错误，就要及时修改，在准确无误后，才可正式在 Internet 上发布。在进行功能性测试和完整性测试后，有的还需要掌握整个站点的结构，以备日后的修改。

5. 发布网站

在网站测试没有问题之后，就可以将其发布到 Internet 上。发布网站时一般需要两步，即申请网页空间和上传网页。域名和空间需要通过在专门的域名及空间提供商那里申请，一般是需要付费的，申请域名及空间成功后，则可以根据提供商所要求的方式将网站上传至指定位置。

6. 网站维护

随着网站的发布，我们应根据访问者的建议，不断修改并更新网站中的信息，并从访问者的角度出发，进一步完善网站。这时网站建设工作又返回到了流程中的第一步，这样周而复始，就构成了网站的维护过程。

3.3.2 确定网站的类型

网站的种类繁多，根据不同的分类方式可以将网站分成不同的类型。

1. 根据网站的规模及专业程度分类

（1）专业型网站

专业型网站一般是由公司、企业、政府机构和社会团体等经营的专业网站，这类网站具有鲜明的特点，能够提供多种服务，具有较高的社会价值和效益。



（2）个人网站

个人网站是个人建立的网站，以展示自我，介绍自身兴趣、爱好和专长为主要目的的小型网站。

2. 根据网站的主题分类

（1）综合性网站

综合性网站相对来说内容比较丰富，涉及的信息量比较大，包含各种方面的信息，如新浪网（http://www.sina.com.cn）、搜狐网（http://www.sohu.com）、网易网（http://www.163.com）、TOM 网（http://www.tom.com/）等都属于综合性网站。

（2）新闻传媒类网站

新闻传媒类网站主要是发布新闻资讯，包括国内外新闻事件、军事、政治资讯等，还可以是各个新闻媒体的门户网站，如新华网（http://www.xinhuanet.com）、人民网（http://www.people.com.cn）、CCTV（http://www.CCTV.com.cn）和凤凰网（http://www.ifeng.com/）等。

（3）政治政策类网站

政治政策类网站一般包括国家政府及各级职能部门的门户网站，如中国中央人民政府网站（http://www.gov.cn/）、教育部网站（http://www.moe.gov.cn/）、商务部网站（http://www.mofcom.gov.cn/）和河南省教育厅网站（http://www.haedu.gov.cn/）等。

（4）教育科技类网站

教育类网站一般包括与教育相关的网站以及各个学校的网站，如教育科研网（http://www.edu.cn）、清华大学网站（www.tsinghua.edu.cn/）、河南公务员考试网（http://www.hnaa.com.cn/）和大学四六级考试官网（http://www.cet.edu.cn/）等。

（5）商业财经类网站

商业财经类网站包括商品交易的网站和提供财经资讯以及分析的专业网站，如和讯网（http://www.hexun.com/）、中国财经信息网（http://www.cfi.net.cn/）、至诚财经网（http://www.zhicheng.com/）、中商贸易网（http://www.cnebiz.net/）、淘宝网（http://www.taobao.com/）和支付宝（https://www.alipay.com/）等。

（6）娱乐休闲类网站

娱乐休闲类网站包括娱乐新闻、音乐欣赏、视频点播以及网络游戏等内容，如中国娱乐网（http://www.67.com/）、九酷音乐网（http://www.9ku.com/）、优酷网（http://www.youku.com/）、土豆网（http://www.tudou.com/）和 4399 小游戏网（http://www.4399.com/）等。

（7）IT 信息类网站

IT 信息类网站包括 IT 资讯和行情报价，涉及电脑、手机、数码产品、软件下载等，如太平洋电脑网（http://www.pconline.com.cn/）、华军软件园（http://www.onlinedown.net/）、电脑之家（http://www.pchome.net/）、天极网（http://www.yesky.com）、中关村在线（http://www.zol.com.cn）和泡泡网（http://www.pcpop.com/）等。

（8）搜索类网站

搜索类网站是应用最为广泛的网站，提供各种信息的搜索，让用户能够很快查找到自己想要的信息，如百度（http://www.baidu.com/）、搜狗（http://www.sogou.com/）、有道

（http://www.youdao.com/）、谷歌（http://www.google.com/）等都属于搜索类网站。

（9）社区类网站

社区类网站主要是为用户提供发表言论的公共平台，如西祠胡同（http://www.xici.net/）、天涯社区（http://www.tianya.cn/）和猫扑网（http://www.mop.com/）等。

（10）文学艺术类网站

文学艺术类网站包含的内容很多，包括文学、书法、雕塑、摄影和收藏等，如 99 艺术网（http://www.99ys.com/）、雅昌艺术网（http://www.artron.net/）、古榕树下原创文学网站（http://www.enjoybar.com/）和起点中文网（http://www.qidian.com/）等。

（11）健康与医药类网站

健康医药类网站一般包括健康资讯和医药资讯等相关内容，也包括各大医院的门户网站，如三九健康网（http://www.39.net/）、21 健康网（http://www.21jk.com.cn/）、导药网（http://www.daoyi.com/）和北京协和医院网站（http://www.pumch.ac.cn/）等。

3.3.3 定位网站的主题

网站主题的确定，需要考虑网站的定位以及网站的目标用户。一个网站必须要有明确的主题，然后围绕主题设计相关的页面。

对于个人网站而言，在选择主题时应注意以下几个问题。

1. 主题选材要小而精

个人网站的主题定位要小，内容要精，不可能包罗万象，包含的内容过多，会使网站显得没有特色和主题，各方面内容都显得肤浅。对于网站还有一个最重要的要求就是更新快，因为一个很长时间都没有更新的网站，访问者是没有兴趣光顾的，所以个人网站必须小而精，才能更新迅速，才能有更强的吸引力。

2. 选择自己擅长或者感兴趣的内容

只有对于擅长的或者感兴趣的内容，在设置内容时，才能够条理清楚，在制作时才能够游刃有余。例如，喜欢足球，就可以选择以足球为主题，报道最新战况、球星动态等；喜欢音乐，可选择以音乐为主题，网页内容可以包括音乐知识普及、音乐欣赏及音乐人物介绍等。

由于网站的内容丰富多彩，网站的主题也越来越新颖独特，要想在网站题材上吸引访问者，确实要下一番工夫，好的主题能够很快抓住访问者的注意力，没有主题的网站会被人很快遗忘。

3.3.4 确定网站的栏目和板块

在确定了网站的主题之后，就要考虑如何设计各种栏目和板块，将网站内容和素材更



好地结合。栏目是指网站中主体突出的内容模块，各模块要围绕主题，实现不同功能。栏目及板块在导航部分要清晰地标识出来。

例如，雅昌艺术网除了首页之外，又包括新闻、展览、评论、视频、博客、论坛、尚品、书画、雕塑、摄影、AAC 艺术中国以及雅昌艺术市场监测中心等几个栏目，每个栏目在导航部分都有相应的标示，访问者单击栏目名的超链接，就可以进入相应的板块，浏览更多的内容，如图 3-5 所示。



图 3-5　雅昌艺术网站的栏目划分

栏目实质上是一个网站的大纲索引，因此栏目的划分能够将网站的板块突出出来。在制定栏目时，要仔细考虑，合理安排，尽可能将与网站主题无关的栏目删除，将最有价值的内容列在栏目上，方便访问者的浏览。

板块比栏目的概念要稍大些，每个板块都有自己的栏目，例如，像新浪这样的大型综合门户网站，一般分为新闻、体育、财经、娱乐等很多板块，每个板块下面又分为多个栏目。对于个人网站而言，栏目和板块的概念基本是一致的。

3.3.5　确定网站的整体风格

一个优秀网站都有自己的个性和文化。风格具有抽象性、独特性和人性等特征。风格的抽象性是指站点的整体形象给浏览者的综合感受；风格的独特性是指该网站不同于其他网站的地方；风格的人性体现在通过网站的外表、内容、文字和交流可以概括出一个站点的个性与情绪，是温文尔雅、执著热情、活泼易变，还是放任不羁。

例如，政府部门网站的风格要庄重，如图 3-6 所示；文化教育类的网站要突出学术氛围，应该高雅大方、格调清新，如图 3-7 所示；娱乐网站则可以生动活泼、色彩艳丽，如图 3-8 所示；商务网站要贴近大众，如图 3-9 所示；综合类门户网站要内容丰富、色彩和谐，如图 3-10 所示。



图 3-6　中华人民共和国外交部网站主页



图 3-7　河南工业大学网站主页



图 3-8　酷狗音乐网站主页





图 3-9 慧聪网主页



图 3-10 新浪网主页

风格是抽象的，是指网站给人的整体感受，包括站点的标志、图像、字体、色彩、版面布局、浏览方式、交互性及内容性等诸多因素。风格是独特的，不管采用何种方式设计网站，都要体现站点所特有的风格。

3.3.6 规划网站目录结构和链接结构

一个完整的网站要包含大量的网页以及很多图片、声音、动画等素材，如果这些内容都简单地放在一个文件夹中，将会不便于管理，这就需要在站点根目录下创建多个子文件夹来存放不同层次的页面，这样整个网站就像一棵大树一样，层次清晰、便于管理。



1. 网站的目录结构

网站的目录结构要根据网站的主题和内容来分类规划，不同的栏目要对应不同的目录，在各个栏目目录下也要根据内容的不同将其划分成不同的分目录，如图 3-11 所示。

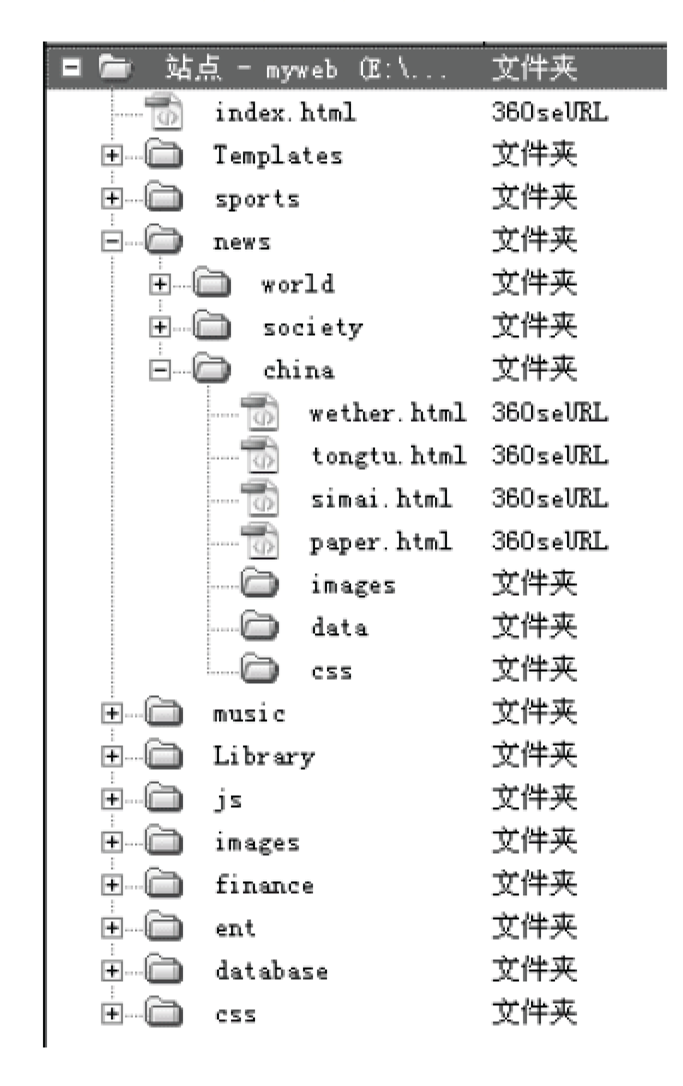


图 3-11　站点目录

在实例中，站点根目录下只建立一个 index.html 网页文件，也就是主页文件，将其他的文件按照栏目内容分别建立子目录，如体育栏目建立 sports 子目录，新闻栏目建立 news 子目录，音乐栏目建立 music 子目录，页面图片放到 images 目录下，库文件放在 Library 目录下，JavaScript 程序文件放在 js 目录下，数据库放到 database 目录下，css 样式表放在 css 目录下等。

每个栏目的目录下可以存放网页文件，也可以继续建立子目录，例如，新闻栏目的 news 目录下，可以进一步建立子目录，如 china 子目录下存放国内新闻的相关文件，world 子目录下存放国际新闻的相关文件，society 子目录下存放社会新闻相关文件等。

另外，每个栏目的目录下，都可以建立独立的 images 目录，用来存放当前栏目中所用的图片素材，这样如需要将某个栏目打包下载，或者将某个栏目整体删除，都很方便。而根目录下的 images 目录一般存放的是首页和一些次要栏目下的图片文件。

注意目录的层次不宜太深，一般不要超过 3 层。另外，给目录起名时要尽量使用能表达目录内容的英文或汉语拼音，这样会更加方便日后的管理与维护。

2. 网站链接结构

网站的链接结构是指页面之间相互链接的拓扑结构，它建立在目录结构基础之上，但可以跨越目录。建立网站链接结构一般有以下两种基本方式。

（1）树状链接结构

树状链接结构一般指的是一对一的结构形式。首页链接指向一级页面，一级页面链接指向二级页面，依此类推。以这样的链接结构浏览时，逐级进入，逐级退出。优点是条理



清晰，访问者明确知道自己在什么位置，不会“迷路”。缺点是浏览效率低，一个栏目下的子页面到另一个栏目下的子页面，必须绕经首页。

（2）星状链接结构

星状链接结构指的是一对多的结构形式。类似网络服务器的连接，每个页面相互之间都建立有链接，立体结构像东方明珠电视塔上的钢球。这种链接结构的优点是浏览方便，随时可以到达自己喜欢的页面。缺点是链接太多，容易使浏览者“迷路”，弄不清自己在什么位置、已经看了多少内容。

这两种基本结构都只是理想方式，在实际的网站设计中，总是将这两种结构混合起来使用，希望浏览者既可以方便、快速地到达自己需要的页面，又可以清晰地知道自己的位置。所以，最好的方法是：首页和一级页面之间用星状链接结构，一级和二级页面之间用树状链接结构。

3.4　小　　结

通过本章学习，掌握在设计网页时应注意的设计原则，学会运用网页制作工具（如 FrontPage、Dreamweaver、Photoshop、Fireworks、AutoCAD、CorelDRAW、Flash、3D Studio Max、Maya、ASP、PHP、JSP 等软件）为网站设计出具有多媒体效果的网页供浏览者观赏。此外，还需要进一步掌握网站设计流程、网站类型选择、网站栏目和板块、网站整体风格等方面的基础知识。

3.5　思　考　题

1．网页包括哪些基本元素？
2．网页设计原则是什么？
3．常见的网页制作软件有哪些？
4．使用 Flash 软件制作一个电子表。
5．使用 Photoshop 软件制作一个校徽。
6．规划一个班级网站，考虑网站类型、主题、栏目、板块、风格及目录结构。

第 4 章　Dreamweaver CS4 工具

第 3 章介绍了网页制作软件，其中 Dreamweaver 是目前网页设计师最青睐的工具之一。因此，本章详细介绍 Dreamweaver CS4 软件的功能，其内容包括 Dreamweaver CS4 的安装、规划、创建和管理站点，创建站点文档，文本的输入和编辑，图像处理，建立网页链接，表格处理，CSS 样式表，嵌入表单元素，添加多媒体元素，框架的使用，站点的整理维护与上传。

4.1　Dreamweaver CS4 概述

4.1.1　Dreamweaver CS4 简介

Dreamweaver CS4 是 Adobe 公司推出的最新版本的网页设计软件，是一款可视化建立 Web 站点和应用程序的专业编辑工具，用于对 Web 站点的页面及程序进行开发和设计。因其功能强大、操作方便、支持多种浏览器并可建立跨平台网页的特点，备受广大网页设计师的青睐，使用时设计师可以根据自己的习惯切换工作环境。Dreamweaver 具备的可视化编辑功能，可使设计者快速创建专业的 CSS 样式的页面，并且为习惯手工编码的程序员提供了许多对编码有帮助的工具和功能，使其能够轻松地使用服务器语言生成支持动态数据库的 Web 应用程序。与之前推出的其他版本相比，CS4 版本进一步改善了 CSS 的功能，从而使创建 CSS 及利用 CSS 对网页进行布局更为便利，同时让使用者无须掌握 Java、HTML、XHTML 等专业语言也能快速布局网页和创建网站。

4.1.2　Dreamweaver CS4 的系统要求和安装

打开 Dreamweaver CS4 安装文件所在的位置，找到 Setup.exe 文件，如图 4-1 所示。

双击 Setup.exe 文件，开始进入 Dreamweaver CS4 的安装，首先进行“检查系统配置文件”的安装初始化工作，如图 4-2 所示。

初始化工作结束之后，开始进入“加载安装程序”阶段，如图 4-3 所示。

加载安装程序结束之后，进入欢迎界面，如图 4-4 所示，其中有两个安装选项需要进行选择，一是“我有 Adobe Dreamweaver CS4 的序列号”，如果没有安装序列号，还可以选择第 2 个选项“我想安装并使用 Adobe Dreamweaver CS4 的试用版”，则有 30 天的



试用期。



图 4-1　Dreamweaver CS4 的安装文件

图 4-2　安装初始化工作




图 4-3　加载安装程序

图 4-4　“欢迎”界面

如果拥有一个正确的安装序列号，输入完成后单击“下一步”按钮，如图 4-5 所示。

单击“下一步”按钮之后进入“许可协议”阶段，要求对 Adobe 最终用户许可协议进行选择，如图 4-6 所示。



图 4-5　输入安装序列号

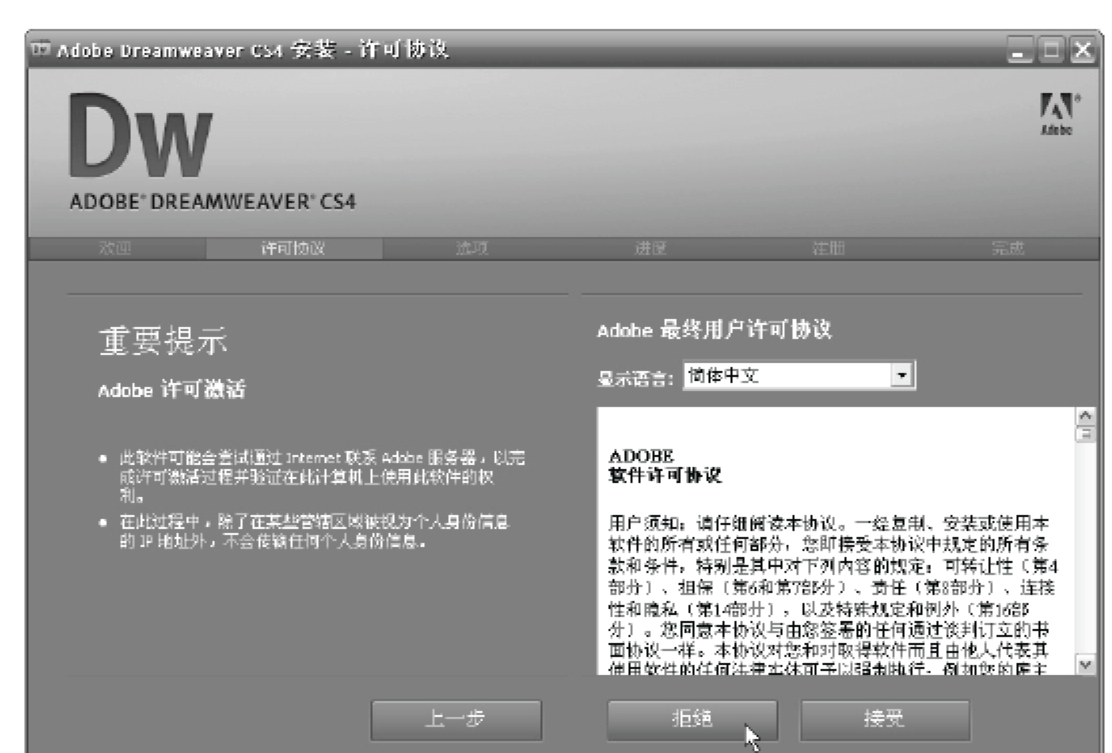


图 4-6　“许可协议”界面

选择“接受”Adobe 软件许可协议之后，进入“选项”界面，可以选择软件的安装语言和安装位置，还可以对安装的组件进行选择，如图 4-7 所示。

对安装语言、位置、组件进行相应的选择之后，单击“安装”按钮就进入了正式安装阶段，会显示整体的安装进度，如图 4-8 所示。



图 4-7 “选项”界面



图 4-8 安装进度界面

正确安装结束之后，会显示安装完成，然后单击“退出”按钮即可退出 Adobe Dreamweaver CS4 的安装界面，如图 4-9 所示。



图 4-9 安装完成界面

4.1.3 启动 Dreamweaver CS4

正确安装 Dreamweaver CS4 软件之后，可以选择“开始”→“程序”→Adobe Dreamweaver CS4 命令启动该软件（或者双击桌面上创建的快捷方式图标），软件的启动界面如图 4-10 所示。

如果在安装时没有输入序列号，软件在启动时会显示还可以继续使用的天数，获得正确序列号之后可以在软件启动时输入，如图 4-11 所示。单击“下一步”按钮可以进入软件的操作界面。



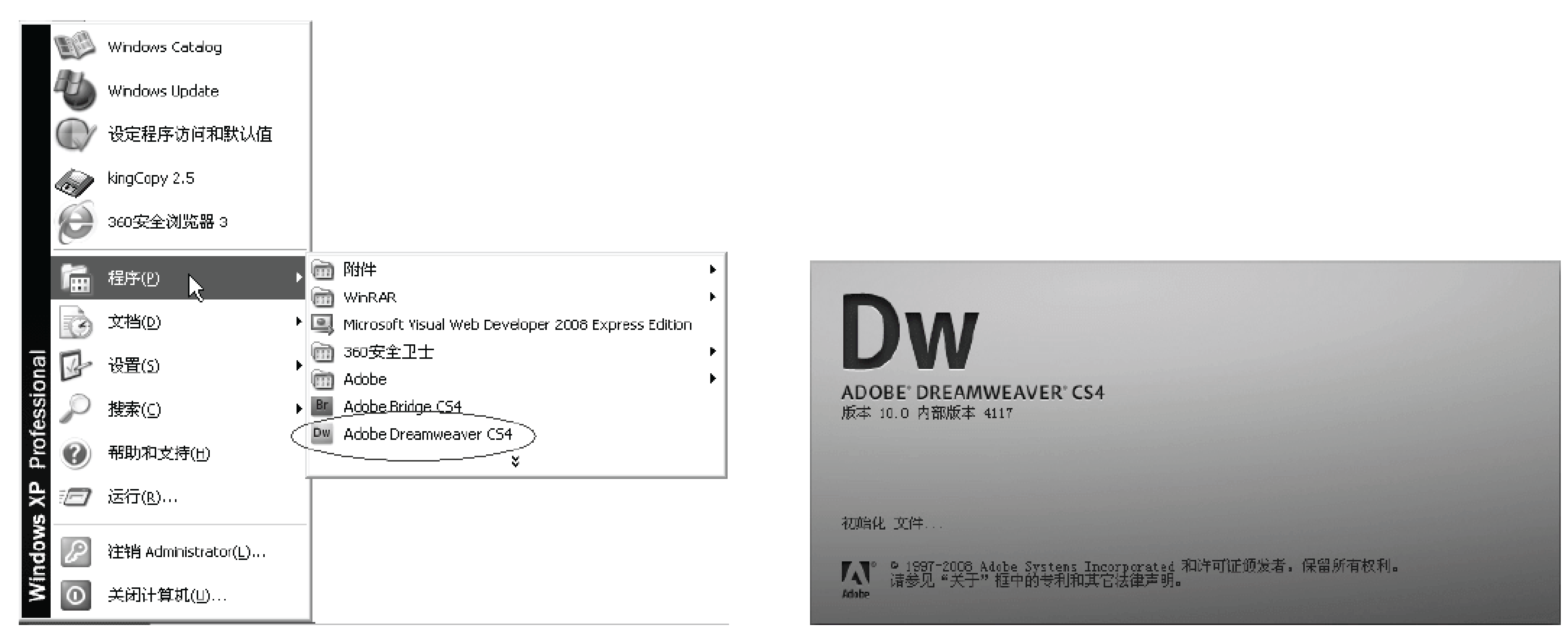


图 4-10　软件的启动界面



图 4-11　试用版本启动界面

4.1.4　Dreamweaver CS4 的操作界面

在 Dreamweaver CS4 的工作区可以查看文档和对象的属性。工作区还将许多常用操作放置于工具栏中，可以快速更改文档。在 Windows 操作系统中，Dreamweaver 提供了一个将全部元素置于一个窗口中的集成布局。在集成的工作区中，全部窗口和面板都被集成到一个更大的应用程序窗口中。

Dreamweaver CS4 的工作界面与 Dreamweaver 以前的版本有所差别，主要由应用程序栏、文档工具栏、文档窗口、标签选择器、属性面板和面板组等部分组成，“插入”栏整合在面板组中，如图 4-12 所示。

（1）欢迎屏幕

欢迎屏幕用于打开最近使用过的文档或创建新文档。还可以在欢迎屏幕中通过产品介绍或教程了解关于 Dreamweaver 的更多信息，如图 4-13 所示。

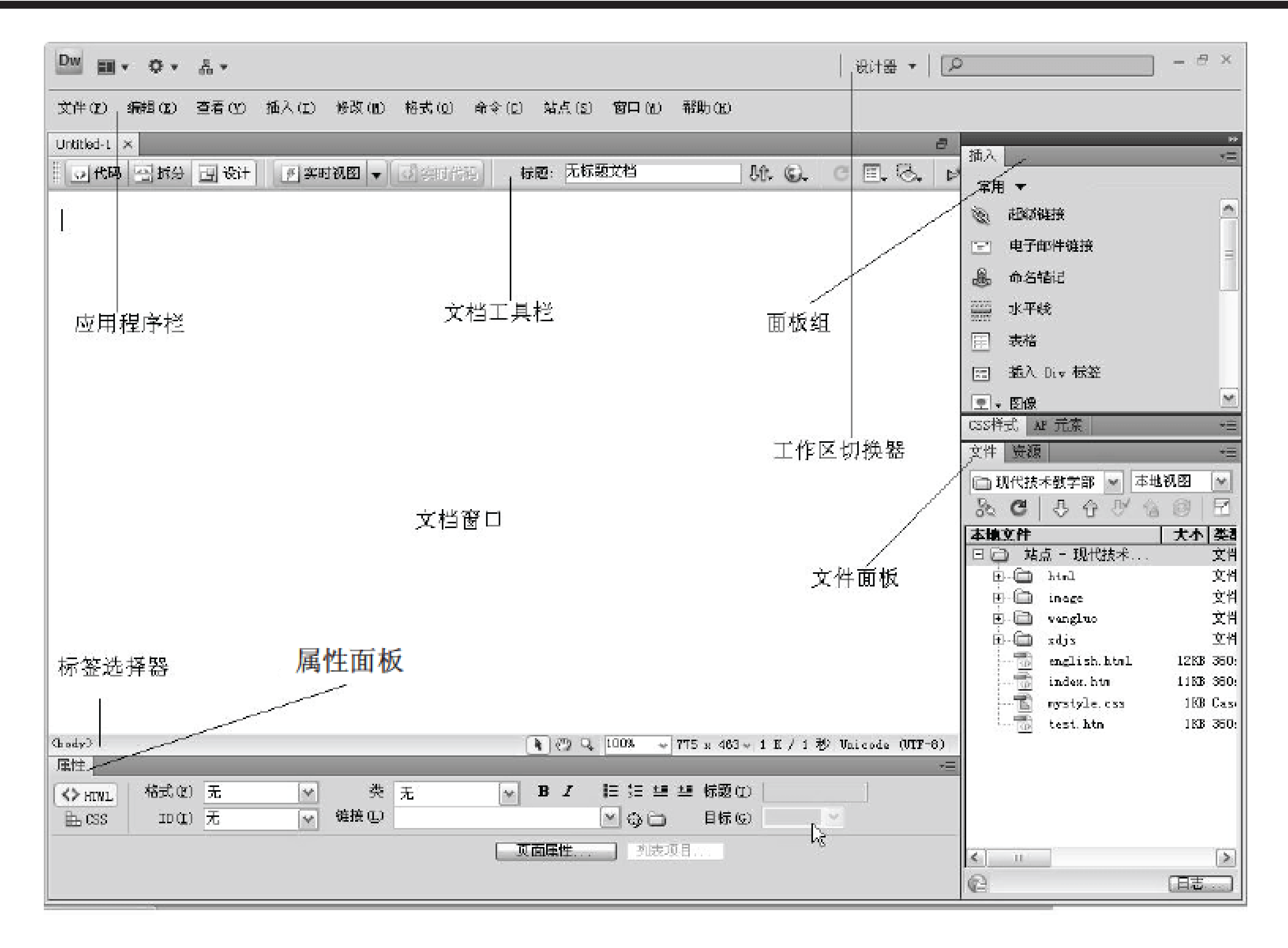


图 4-12　Dreamweaver CS4 操作界面



图 4-13　欢迎屏幕

（2）应用程序栏

应用程序窗口顶部包含一个工作区切换器、几个菜单（仅限 Windows）以及其他应用程序控件，如图 4-14 所示。

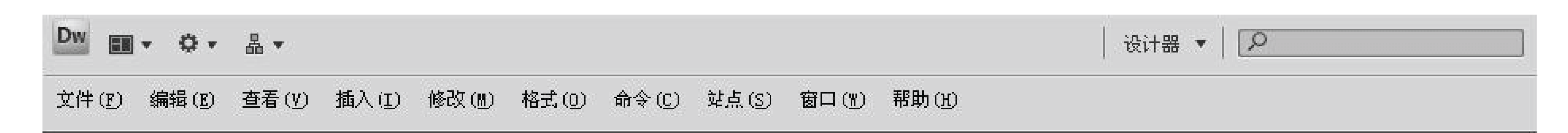


图 4-14　应用程序栏

（3）文档工具栏

使用文档工具栏包含的按钮可以在文档的不同视图之间快速切换。工具栏中还包含一些与查看文档、在本地和远程站点间传输文档有关的常用命令和选项，并提供各种文档窗



口视图的选项（如“设计”视图和“代码”视图）、各种查看选项和一些常用操作（如在浏览器中预览），如图 4-15 所示。



图 4-15　文档工具栏

（4）标准工具栏

标准工具栏在默认工作区布局中不显示，其中包含一些按钮，可用于执行“文件”和“编辑”菜单中的常见操作，如新建、打开、在 Bridge 中浏览、保存、全部保存、打印代码、剪切、复制、粘贴、撤销和重做。若要显示标准工具栏，可选择“查看”→“工具栏”→“标准”命令，弹出的标准工具栏如图 4-16 所示。



图 4-16　标准工具栏

（5）编码工具栏

编码工具栏包含可用于执行多种标准编码操作的按钮，如折叠和展开所选代码、高亮显示无效代码、应用和删除注释、缩进代码、插入最近使用过的代码片断等。编码工具栏垂直显示在文档窗口的左侧，仅当显示“代码”视图时才可见，如图 4-17 所示。

图 4-17　编码工具栏

（6）样式呈现工具栏

样式呈现工具栏中包含一些按钮，如果使用依赖于媒体的样式表，则可使用这些按钮查看用户的设计在不同媒体类型中的呈现效果。还包含一个允许用户启用或禁用层叠式样式表（CSS）样式的按钮，默认为隐藏状态，如图 4-18 所示。



图 4-18　样式呈现工具栏

（7）文档窗口

文档窗口用于显示用户当前创建和编辑的文档。

（8）属性面板

属性面板可以检查和编辑当前选定页面元素（如文本和插入的对象）的最常用属性。其中的内容根据选定元素的不同会有所不同。例如，如果选择页面上的一个图像，则属性面板将显示该图像的属性（如图像的文件路径、宽度和高度、周围的边框等）。

属性面板在编码器工作区布局中默认是不展开的，如图 4-19 所示。



图 4-19　属性面板

（9）标签选择器

位于文档窗口底部的状态栏中，显示环绕当前选定内容的标签的层次结构。单击该层次结构中的任何标签可以选择该标签及其全部内容，如图 4-20 所示。

<body><div><table><tr><td><table><tr><td> 100% 727 x 463 71 K / 11 秒 简体中文(GB2312)

图 4-20　标签选择器（位于状态栏的左侧）

（10）面板组

面板组用于帮助用户监控和修改工作，包括“插入”面板、“CSS 样式”面板和“文件”面板等。若要展开某个面板，则双击即可。

- “插入”面板

“插入”面板包含用于将图像、表格和媒体元素等各种类型的对象插入到文档中的按钮。每个对象都是一段 HTML 代码，允许用户在插入它时设置不同的属性。例如，可以在“插入”面板中选择“表格”选项，以插入一个表格。当然也可以不使用“插入”面板，而使用“插入”菜单来插入对象，如图 4-21 所示。

- “文件”面板

“文件”面板用于管理文件和文件夹，这些文件既可以属于 Dreamweaver 站点，也可以位于远程服务器上。“文件”面板还可使用户访问本地磁盘上的全部文件，非常类似于 Windows 的资源管理器，如图 4-22 所示。



图 4-21　“插入”面板



图 4-22　“文件”面板

在“文件”面板中查看站点、文件或文件夹时，可以更改查看区域的大小，还可以展开或折叠“文件”面板。当折叠“文件”面板时，它以文件列表的形式显示本地站点、远程站点、测试服务器和 SVN 库的内容。在展开时，它会显示本地站点、远程站点、测试服务器或 SVN 库中的其中一个。对于 Dreamweaver 站点，还可以通过更改折叠面板中默认显示的视图（本地站点视图或远程站点）来对“文件”面板进行自定义。



4.2　规划与创建 Dreamweaver 站点

4.2.1　规划站点结构

一般情况下，在 Dreamweaver 中，用户需要首先在本地计算机上创建本地站点，并对站点结构做合理有序的规划，然后添加相关内容，最后将站点上传至 Internet 服务器。

站点结构主要是指网站的目录结构和链接结构，两者应当相互配合，共同搭建网站的“骨架”。

1. 目录结构

网站中所有的内容通常保存在 Web 服务器的某个目录中，在规划站点时，应该事先在计算机硬盘上建立一个文件夹作为工作的起点，然后在文件夹下新建若干个子文件夹组成合理的目录结构，以便将不同类型的文件存放在站点中。

目录结构本身对于网站的浏览效果并无较大影响，但对于网站建设中的上传和维护，以及将来内容的扩充和移植都有着重要的影响。

规划网站的目录结构，应遵循以下基本原则。

（1）根据栏目设置建立子目录

对于综合性网站而言，如果把所有的文件都保存在根目录，将会导致文件管理混乱的严重问题。应该根据网站的栏目内容建立多个子目录，在每个子目录保存该栏目的所有文件。在根目录下一般只保存网站首页文件及其他重要文件。如果这个栏目的内容特别多，那么应该根据其内容，再在二级栏目下设立三级栏目，并根据三级栏目建立相应的文件子目录，依此类推。但需要注意目录的层次不要太多，为了维护方便，目录的层次一般不要超过 3 层。

（2）为每个栏目建立独立的图片目录

通常，在每个网站的根目录下都应设置一个图片目录（images），用来存放网页中的图片。如果把网站中所有的图片都放在这个目录，以后很可能会混淆不清。因此，最好是在每个栏目的子目录中建立一个独立的 images 目录，以存放该栏目中的所有图片，而根目录下的 images 目录则只存放首页中的图片。

（3）采用正确的命名方式

网站中所有的子目录和文件应避免使用中文目录和中文文件名，使用中文目录可能对网址的正确显示造成困难。文件名不要太长，尽管 Web 服务器支持长文件名，但是太长的文件名不便于记忆，也不便于管理。另外，文件名尽量意义明确，如栏目名称的汉语拼音或者英文名，以便于记忆和管理。

本章以“现代技术教学部”网站为例进行介绍，其网站栏目设置、网站目录结构和站点本地视图如图 4-23、图 4-24 和图 4-25 所示。

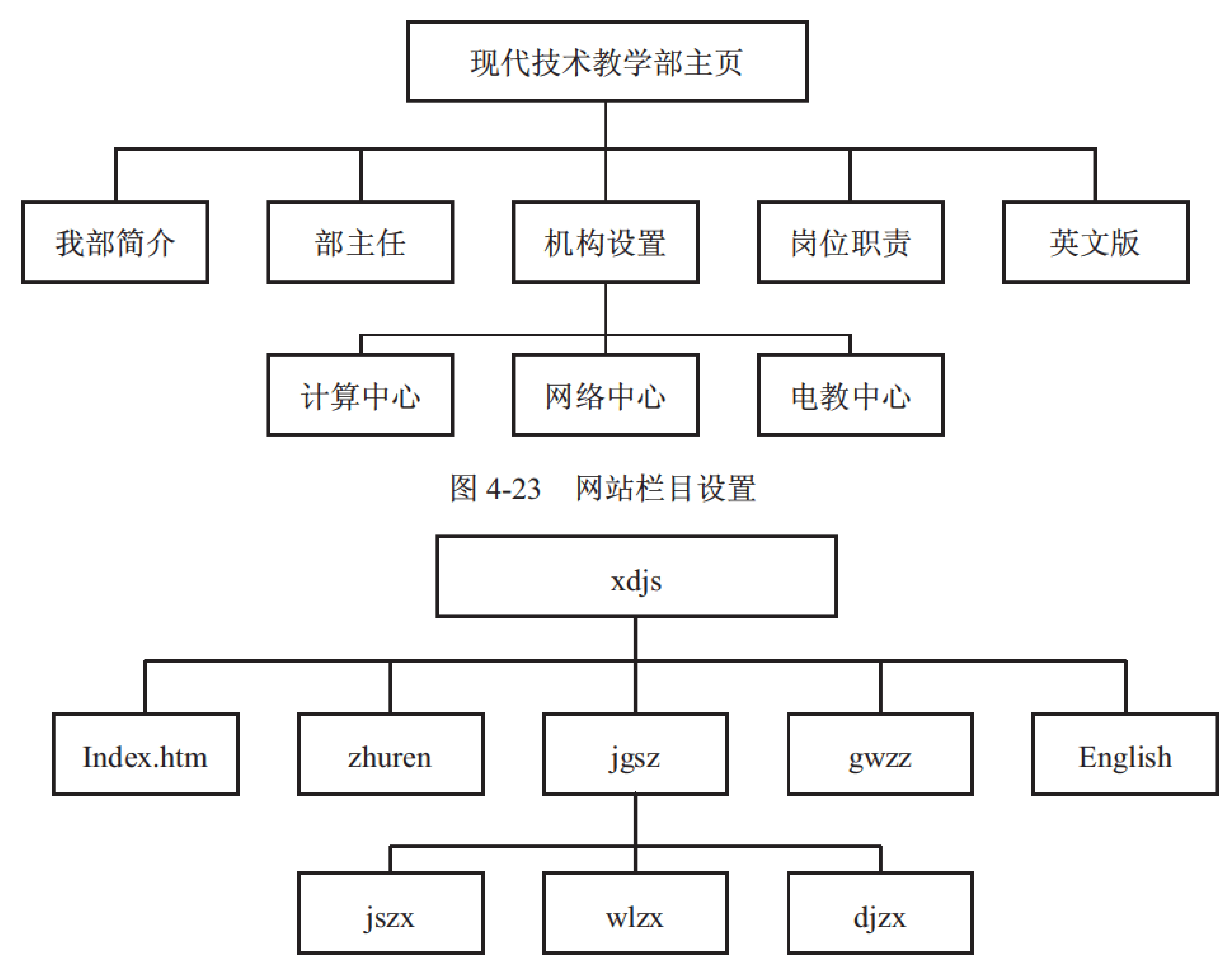


图 4-23　网站栏目设置

图 4-24　网站目录结构

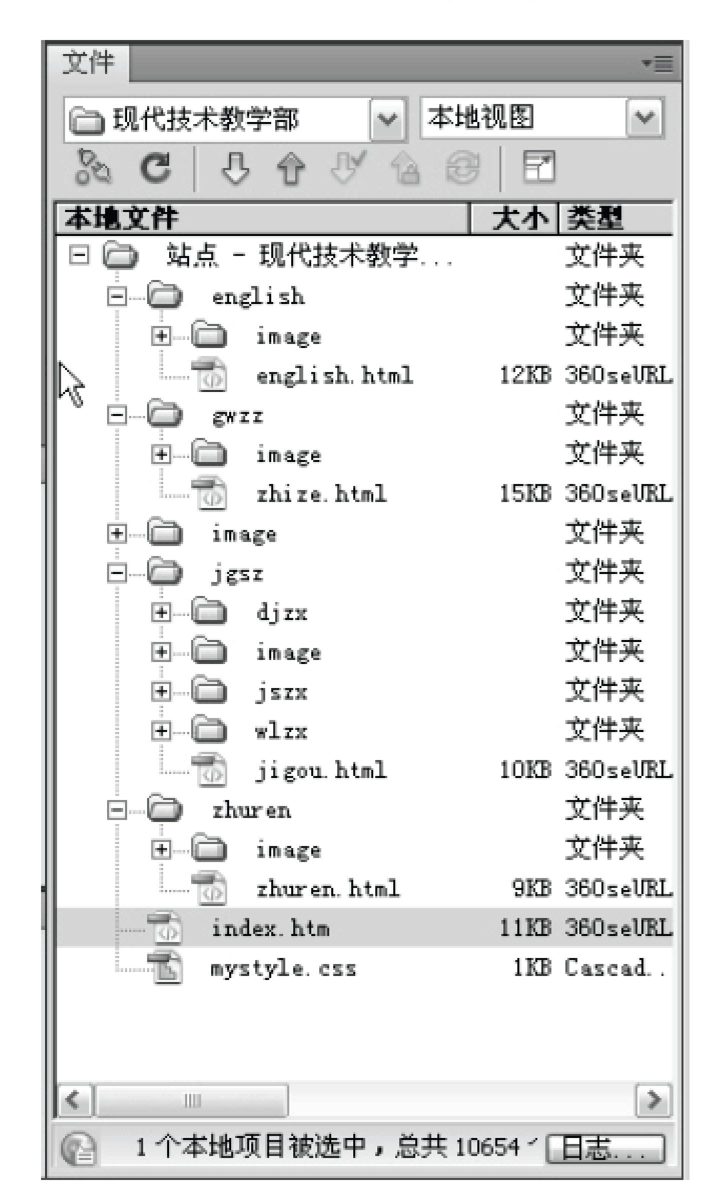


图 4-25　站点本地视图

2．链接结构

网站的链接结构是指网页之间相互链接的关系。正确的链接结构能让浏览者方便地访问网站中的所有内容，而不会漏掉一些重要页面。网站的链接结构建立在目录结构的基础



之上，但又可以超越目录，它主要有两种基本方式：树状结构和星状结构。

在树状结构中，以主页作为网站的“主干”，首页链接指向一级页面，一级页面链接指向二级页面。这样的链接结构条理清晰，但浏览效率低，一个栏目下的子页面到另一个栏目下的子页面，必须绕经主页。

在星状结构中，每个网页相互之间相互链接，浏览方便，浏览者可以随时到达其他页面。但是由于链接太多，容易使浏览者“迷路”。在实际的网站设计中，总是将这两种结构混合起来使用，让浏览者既可以方便快速地达到自己需要的页面，又清晰地了解当前访问位置。所以，最好的办法是首页和一级页面之间用星状链接结构，一级和二级页面之间用树状链接结构。现代技术教学部主页的首页面（index.htm）设置的链接如图 4-26 所示。

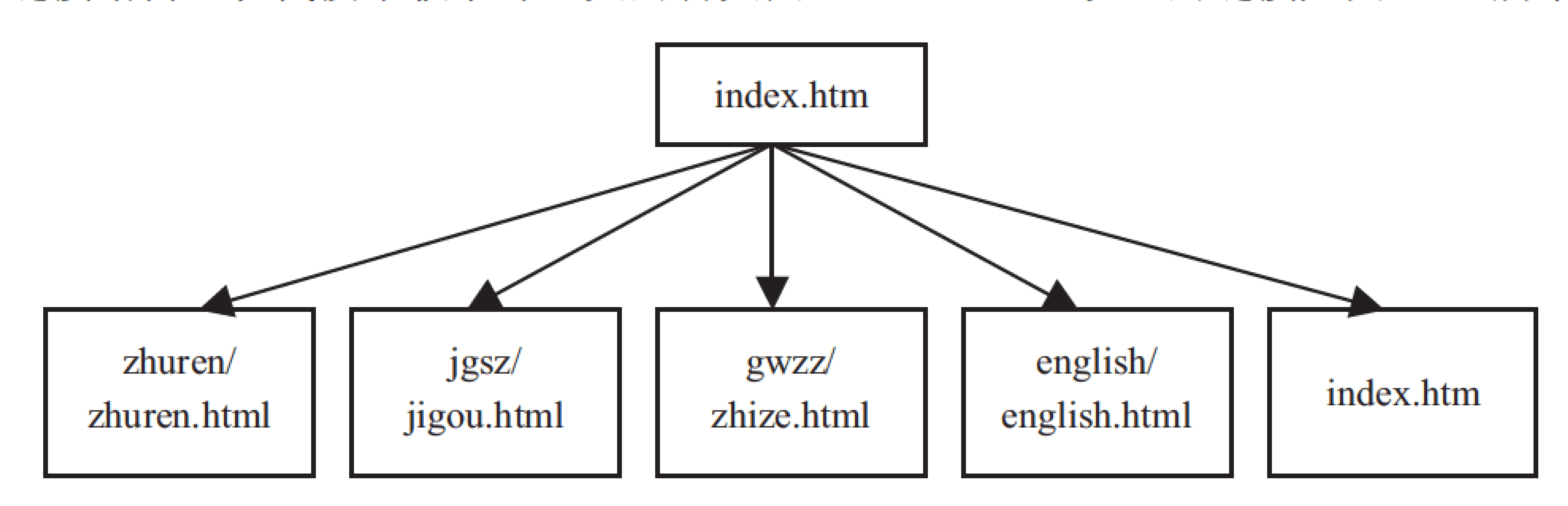


图 4-26 首页面 index.htm 设置的链接

4.2.2 建立本地站点

在 Dreamweaver 中，术语“站点”指属于某个 Web 站点的文档的本地或远程存储位置。Dreamweaver 站点提供了一种方法，使用户可以组织和管理所有的 Web 文档，将站点上传到 Web 服务器，并跟踪和维护站点链接以及管理和共享文件。在开始网页设计制作之前，首先应定义一个站点，以充分利用 Dreamweaver CS4 强大的站点管理功能。

定义 Dreamweaver 站点，只需设置一个本地文件夹。若要向 Web 服务器传输文件或开发 Web 应用程序，还必须添加远程站点和测试服务器信息。

Dreamweaver 站点由 3 个部分（或文件夹）组成，具体取决于开发环境和所开发的 Web 站点类型。

（1）本地根文件夹：存储用户正在设计和处理的文件。Dreamweaver 将此文件夹称为“本地站点”。此文件夹通常位于本地计算机上，但也可能位于网络服务器上。

（2）远程文件夹存储：用于测试、生产和协作等用途的文件。Dreamweaver 在“文件”面板中将此文件夹称为“远程站点”。远程文件夹通常位于运行 Web 服务器的计算机上，包含用户从 Internet 访问的文件。

通过本地文件夹和远程文件夹的结合使用，用户可以在本地硬盘和 Web 服务器之间传输文件，可以帮助用户轻松地管理 Dreamweaver 站点中的文件，还可以在本地文件夹中处理文件，希望其他人查看时，再将它们发布到远程文件夹。

（3）测试服务器文件夹：Dreamweaver 在其中处理动态页的文件夹。

1. 建立本地站点

利用 Dreamweaver CS4 提供的站点创建向导，用户可以轻松地建立本地站点，具体操作步骤如下：

（1）选择“站点”→“新建站点”命令，如图 4-27 所示，或者选择“管理站点”命令，在弹出的如图 4-28 所示的对话框中单击“新建”按钮，然后在弹出的菜单中选择“站点”命令，弹出如图 4-29 所示的“未命名站点 1 的站点定义为”对话框。

（2）在如图 4-29 所示的对话框中的“您打算为您的站点起什么名字？”文本框中输入站点名称，如“xdjs”，在“您的站点的 HTTP 地址（URL）是什么？”文本框中输入所创建站点的 HTTP 地址，然后单击“下一步”按钮。



图 4-27　新建站点



图 4-28　“管理站点”对话框

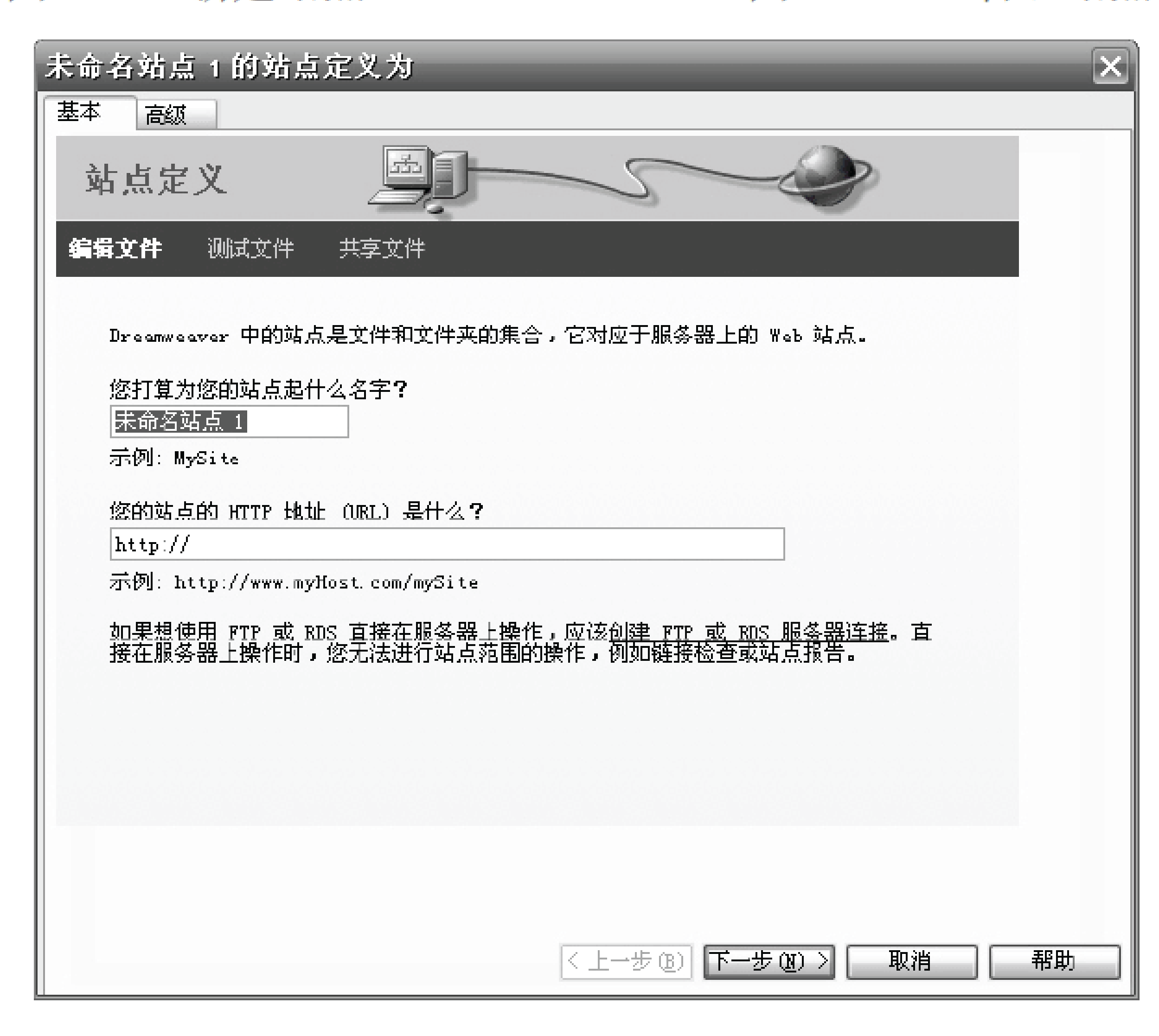


图 4-29　“未命名站点 1 的站点定义为”对话框

注意： 站点名称不是网站的名称，这里说的站点名称实际上就是创建站点时存放站点的文件夹的名称（例如，把站点文件存放在 F:\xdjs\中，站点名称就是 xdjs），而网站名称是根据网站的主题为网站起的名字（新浪网站中的“新浪”就是网站名称）。书写站点名称时需要注意避免使用汉字当作站点名称，这条规则同样适用于网页的名称，文件名中可使用空格符，但不允许使用“!”、““”、“？”、“\ ”、“*”、“<”、“>”。



（3）在弹出的如图 4-30 所示的对话框中，根据网站是否需要服务器技术及何种脚本语言来选择服务器技术，例如，选中“否，我不想使用服务器技术”单选按钮，然后单击“下一步”按钮。



图 4-30　“xdjs 的站点定义为”对话框

注意：服务器技术涉及动态页面的编程和数据库连接问题，可参考本书的其他相关章节内容。本章是以创建静态页面为例，所以选中“否，我不想使用服务器技术”单选按钮。

（4）在弹出的如图 4-31 所示的对话框中选中“编辑我的计算机上的本地副本，完成后再上传到服务器（推荐）”单选按钮。

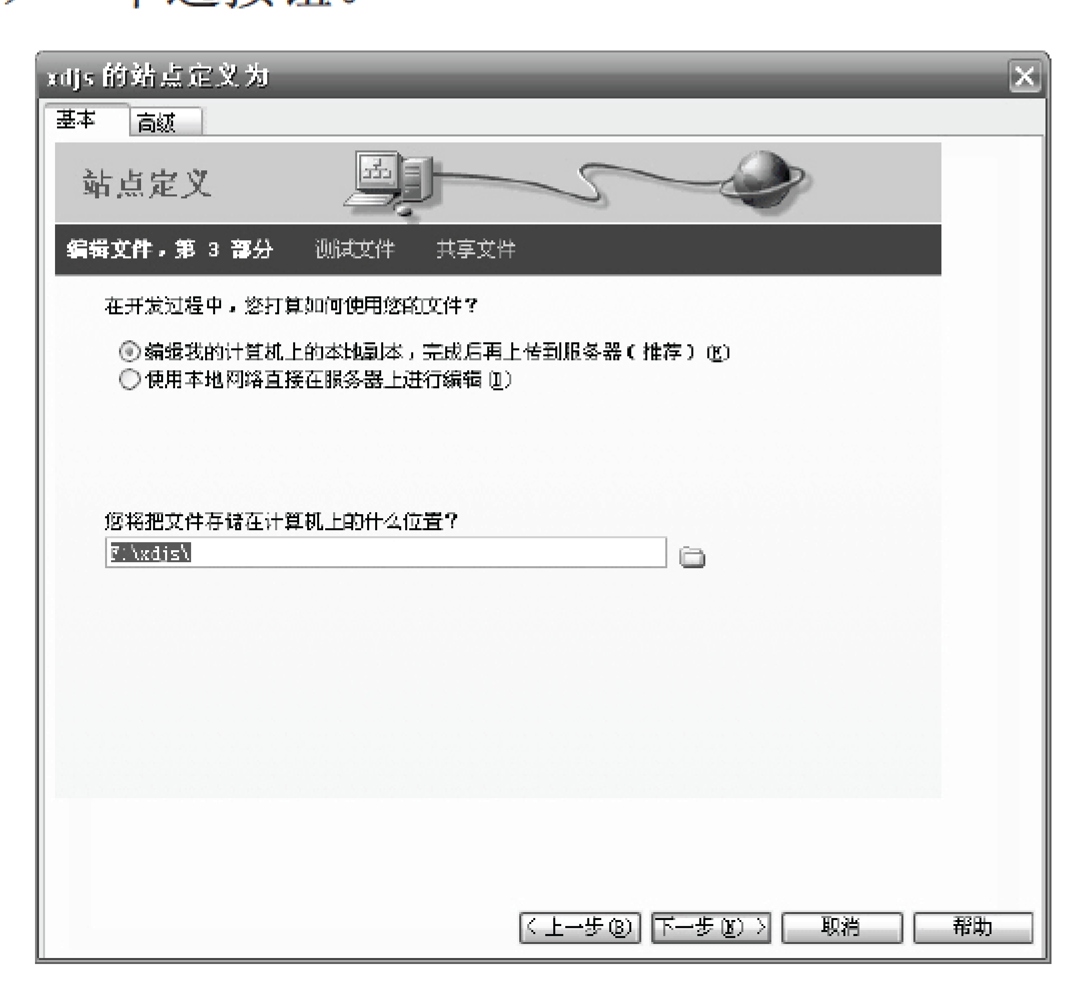


图 4-31　选中“编辑我的计算机上的本地副本，完成后再上传到服务器（推荐）”单选按钮

如果用户还没有专门建立存放站点的文件夹，最好在图 4-31 的对话框中输入站点保存的路径（例如，F:\xdjs\指的是硬盘 F 盘中的 xdjs 文件夹，它就是站点的根目录）；如果用户已经建立好存放站点的文件夹，单击“浏览”按钮选择存放站点的文件夹，选择完成后单击“下一步”按钮。

（5）在弹出的如图 4-32 所示的对话框中，为 Dreamweaver 选择一种访问远程文件夹的方法。该对话框所显示内容将随着下拉列表框中选择内容的改变而改变，访问远程文件夹的方法可设为 FTP、WebDAV、SourceSafe 数据库、RDS 或本地/网络等，这里选择“无”选项，即选择本地站点作为远程服务器。

（6）单击“下一步”按钮，将弹出如图 4-33 所示的对话框，在其中可以看到从创建站点的第一步到最后一步过程中选择的所有信息。如信息有误，单击“上一步”按钮重新设置；如确认无误，单击“完成”按钮结束站点创建。



图 4-32　选择站点作为远程服务器

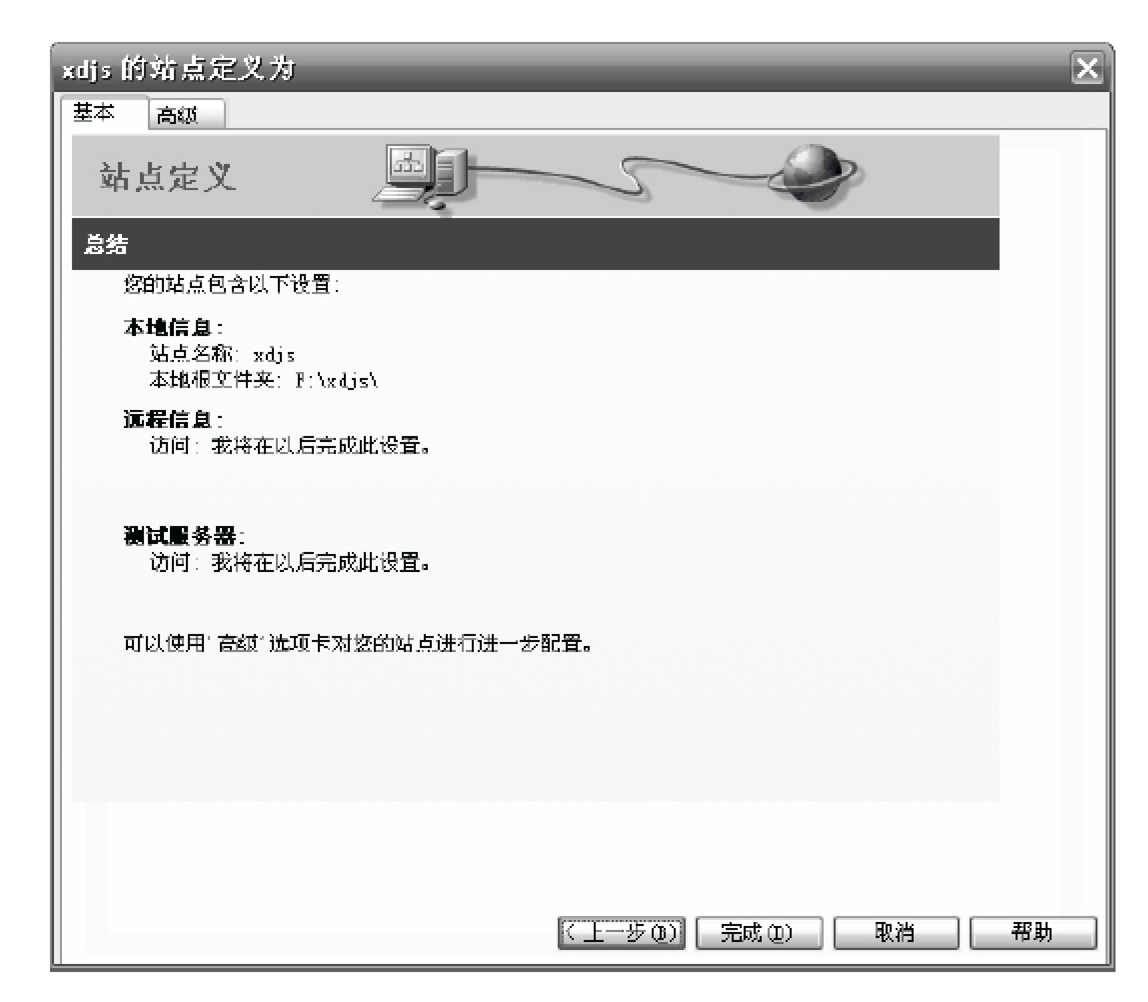


图 4-33　总结对话框

2．本地站点的管理

站点创建完成之后，还可以在如图 4-34 所示的“管理站点”对话框中对创建的一个或多个站点进行相应的管理工作。

（1）在“管理站点”对话框中可以看到所有已经创建好的站点，还可根据需要建立新的站点。

（2）单击“编辑”按钮，可以对所选择的某个站点的相应参数进行重新配置和选择。

（3）单击“复制”按钮，可以创建所选择的站点的副本，创建的副本将出现在站点列表中，如图 4-35 所示。



图 4-34　“管理站点”对话框



图 4-35　复制站点

（4）单击“删除”按钮，将从站点列表中删除所选择的 Dreamweaver 站点及其所有设



置信息，此操作无法撤销。但并不会将站点文件从本地计算机中删除。单击“删除”按钮后，在弹出的如图 4-36 所示的对话框中单击“是”按钮，则从站点列表中删除站点；单击“否”按钮则保留站点名称，然后单击“完成”按钮。

（5）单击“导出”按钮可以将站点设置导出为 XML 文件，并在以后将该文件导入 Dreamweaver 中，可以方便地在各计算机和产品版本之间移动站点或者与其他用户共享设置，如图 4-37 所示。



图 4-36　删除站点



图 4-37　导出站点

导出站点的步骤如下：

① 选择“站点”→“管理站点”命令。

② 选择要导出设置的一个或多个站点，然后单击“导出”按钮。

- 若要选择多个站点，可按住 Ctrl 键单击每个站点。
- 若要选择某一范围的站点，可按住 Shift 键单击该范围中的第一个和最后一个站点。

③ 对于要导出的每个站点，可浏览要保存站点的位置，然后单击“保存”按钮，Dreamweaver 会将每个站点的设置保存为扩展名为.ste 的 XML 文件。

（6）单击“导入”按钮，浏览并选择.ste 文件中定义的且要导入其设置的一个或多个站点。导入该站点设置之后，站点名称会出现在“管理站点”对话框中。

4.3　创建站点文档

新创建的站点是空的，没有任何文件，用户需要在其中创建并编辑自己的网页文件，还可以对其进行各种管理。

4.3.1　创建新文档

创建文件的操作步骤如下：

（1）选择“窗口”→“文件”命令，打开“文件”面板，如图 4-38 所示。

（2）选中需要创建文件的文件夹，单击鼠标右键，在弹出的快捷菜单中选择“新建文件”命令，在“文件”面板中将生成一个名称为 Untitled-1.html 的文件，如图 4-38 所示。

创建新文档也可以采用如下步骤：

（1）选择“文件”→“新建”命令，弹出如图 4-39 所示的“新建文档”对话框，默认选择“空白页”选项卡。



图 4-38 “文件”面板

图 4-39 “新建文档”对话框

（2）从右侧的“页面类型”列表中选择 HTML 选项，单击“创建”按钮，Dreamweaver 即创建一个新的 HTML 文件，并展开工作区界面，如图 4-40 所示。

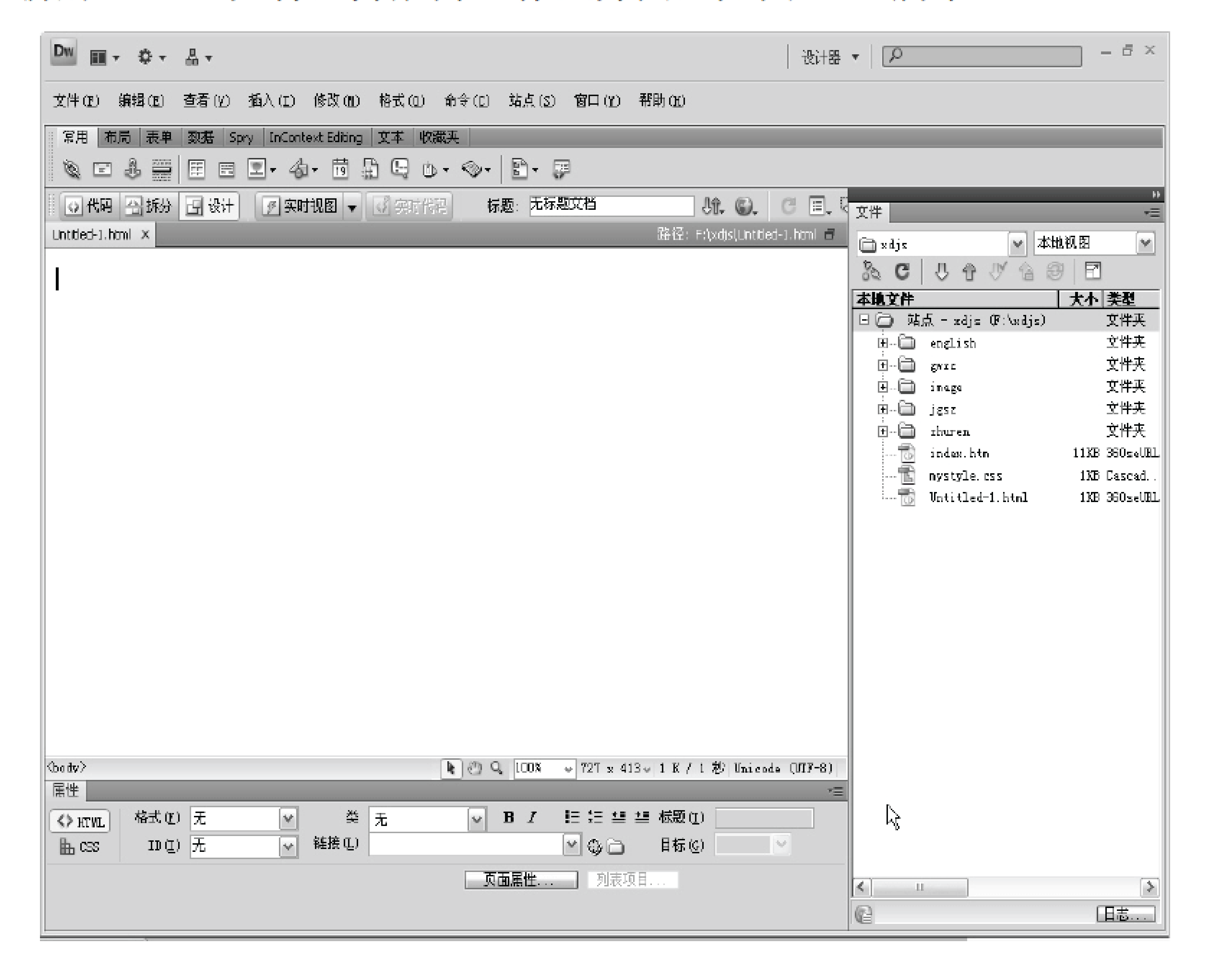


图 4-40 新建网页



此时网页的默认文件名为 Untitled-1.html，新建的 HTML 网页生成如下代码段：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/ xhtml1/
DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>无标题文档</title>
</head>
<body>
</body>
</html>
```

4.3.2　文档的设置

对于站点中创建好的各种文档，还可以通过“文件”面板进行相应的管理。

（1）移动文件

打开“文件”面板后，选中需要移动的文件，按住鼠标左键不放，拖动该文件到面板的其他位置，即可移动文件。由于文件的位置发生了变化，其中的链接信息也可能相应发生变化。

（2）复制文件

打开“文件”面板后，选中需要复制的文件，单击鼠标右键，在弹出的快捷菜单中选择“编辑”→“复制”命令即可复制文件。

（3）重命名文件

打开“文件”面板后，选中需要重命名的文件，单击鼠标右键，在弹出的快捷菜单中选择“编辑”→“重命名”命令，使该文件的名称变为可编辑状态，输入一个新的名称，按 Enter 键即可重命名文件。

（4）删除文件

打开“文件”面板后，选中需要删除的文件，按 Delete 键，此时系统将弹出一个提示框，提示用户是否要删除文件，单击“是”按钮，该文件即可从本地站点中删除。

4.3.3　设置页面属性

页面标题、背景图像和颜色、文本和链接颜色、边距是每个 Web 文档的基本属性。在 Dreamweaver 中打开一个网页文件后，单击属性面板中的“页面属性”按钮或者选择“修改”→“页面属性”命令，均可打开“页面属性”对话框（如图 4-41 所示）进行设置或更改页面属性。Dreamweaver 将页面属性分为外观（CSS）、外观（HTML）、链接（CSS）、标题（CSS）、标题/编码和跟踪图像 6 个类别。

（1）设置外观（CSS）属性

“页面属性”对话框默认显示“外观（CSS）”设置页面，在其中可设置整个页面的字体、文字的大小、文本颜色、背景颜色、背景图像、重复、左边距、右边距、上边距和下边距等内容，如图 4-41 所示。

（2）设置外观（HTML）属性

选择“外观（HTML）”选项，在右侧的设置界面中可以设置背景颜色、背景图像、链接文本颜色、左边距、上边距、边距宽度和高度等内容。此时设置的属性会导致页面采用 HTML 格式，而不是 CSS 格式，如图 4-42 所示。



图 4-41　“页面属性”对话框

图 4-42　外观（HTML）

（3）链接（CSS）

选择“链接（CSS）”选项，在右侧的设置界面中可以设置链接字体、大小、颜色，还可以设置变换图像链接、已访问链接、活动链接的颜色以及下划线的样式，如图 4-43 所示。

（4）标题（CSS）

选择“标题（CSS）”选项，在右侧的设置界面中可以设置标题字体的属性。这里的标题指的并不是页面的标题内容，而是可以应用在具体文章中各级不同标题上的字体样式，如图 4-44 所示。

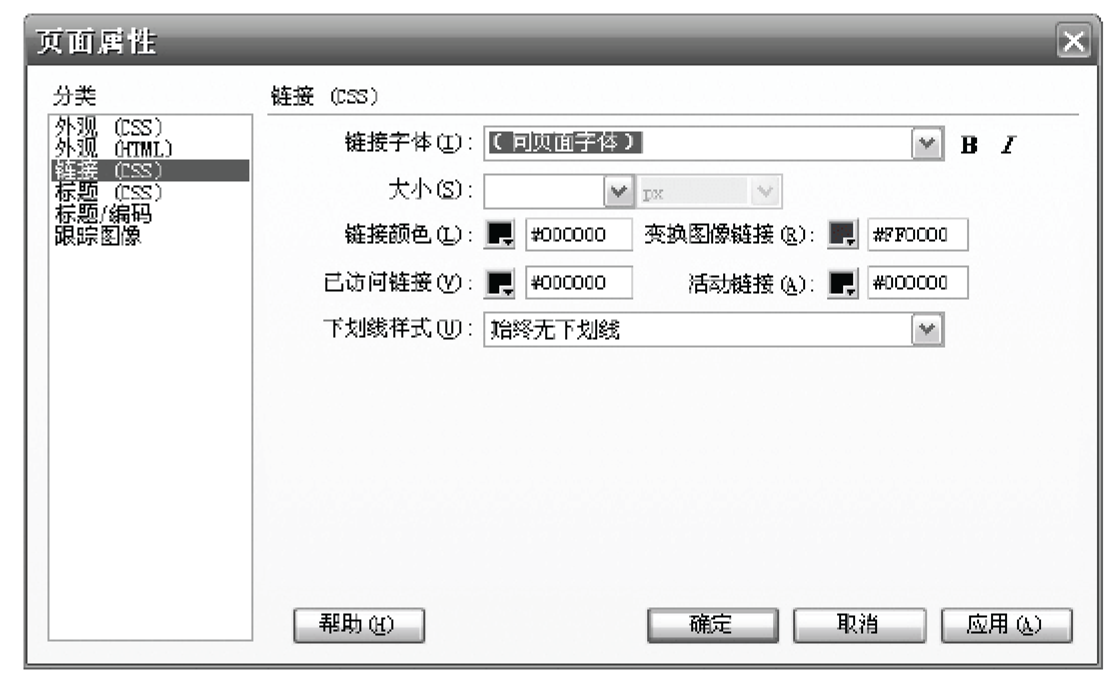


图 4-43　链接（CSS）

图 4-44　标题（CSS）

（5）标题/编码

选择“标题/编码”选项，在右侧的设置界面中可以设置页面的标题和编码的属性。“标题”文本框中输入的内容就是在 IE 浏览器的标题栏看到的页面标题。“文档类型”下拉列



表框用于指定文档类型。“编码”下拉列表框用于指定文档中字符所用的编码。简体中文网站通常采用的是简体中文（GB2312），如图 4-45 所示。

（6）跟踪图像

选择“跟踪图像”选项，在右侧的设置界面中可以插入一个图像文件，并在设计页面时使用该文件作为参考。“跟踪图像”文本框用于指定在复制设计时作为参考的图像，该图像只供参考，当文档在浏览器中显示时并不出现。“透明度”滑条确定跟踪图像的不透明度，范围为从完全透明到完全不透明，如图 4-46 所示。



图 4-45　标题/编码



图 4-46　跟踪图像

4.4　文本的输入和编辑

文字是网页中重要的设计元素，虽然声音、Flash 等多媒体元素在网页中的比重越来越大，但是文字依然占据着传递信息的主导地位。因此对于网页设计者而言，了解和掌握网页设计中的文字排版设计就显得尤为重要。

4.4.1　输入文本

1. 插入普通文本

若要向 Dreamweaver 文档添加文本，可以直接在文档窗口中输入文本，也可以剪切并粘贴文本，还可以从其他文档导入文本。在 Dreamweaver 中插入普通文本的方法和在 Word 中插入文本的方法基本相同，有如下几种方法：

- 把鼠标光标停在文档窗口中需要输入文字的位置，切换到合适的输入法输入文字即可。
- 从其他文档中剪切并粘贴所需的文字。打开文字所在文档，选中所需文字后复制到文档窗口中需要的位置。当使用“选择性粘贴”命令时，会弹出“选择性粘贴”对话框，如图 4-47 所示。



图 4-47 “选择性粘贴”对话框

- 从其他文档导入文本。选择“文件”→“导入”→“Word 文档”命令，弹出“导入 Word 文档”对话框，浏览所需的文件或在“文件名”文本框中输入所需文件的名称，最后单击“打开”按钮，如图 4-48 所示。

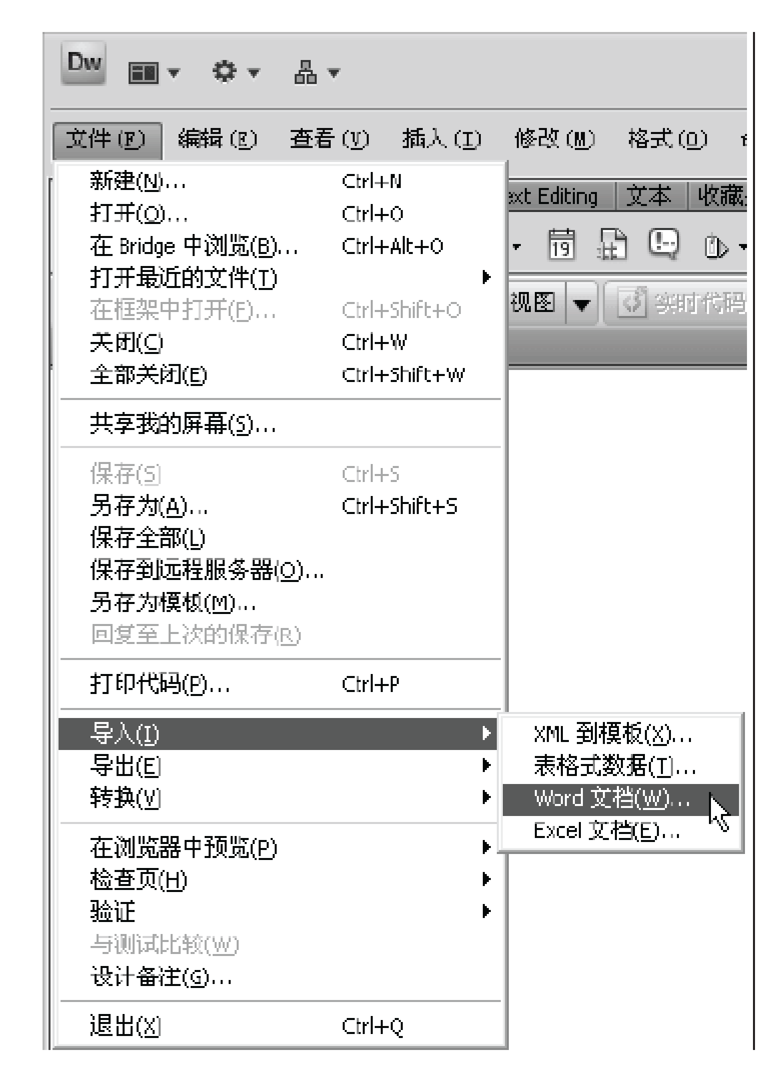




图 4-48 导入“Word 文档”

2. 插入列表文本

使用列表文本可使页面看起来明朗清晰，列表文本分为两种类型：一种是无序列表文本，另一种是有序列表文本。无序列表前多以相同的符号来引导，而有序列表前多以数字或字母来引导。

插入列表文本的方法如下：

（1）单击文本属性面板中的“项目列表”按钮可插入无序列表文本；单击“项目编号”按钮可插入有序列表文本。

（2）在文档窗口中输入文本，按 Enter 键另起一行，列表符号将自动出现。

（3）按 Enter 键，确认完成插入列表文本操作。

4.4.2 设置文本属性

在文档窗口中选中一段文字，其属性面板有以下两种方式：



- HTML 方式。在此面板中设置文本属性采用的是 HTML 格式，而不是 CSS 格式，如图 4-49 所示。



图 4-49　HTML 方式属性面板

- CSS 方式。在此面板中设置文本属性采用的是 CSS 格式，如图 4-50 所示。



图 4-50　CSS 方式属性面板

在两种类型的属性面板中均可以定义文字的格式、字体、样式、字号、颜色、加粗、倾斜和水平对齐等内容。

1．设置文本的格式

利用“格式”菜单下的“段落格式”子菜单可对选中的段落设置格式，利用属性面板中的“格式”下拉列表框也可以选择段落或标题格式，如图 4-51 所示。“标题 1”到“标题 6”分别表示各级标题，应用于网页的标题部分。“标题 1”的字体最大，然后依序慢慢变小，可根据需要套用。进行“标题 1”到“标题 6”的设置对应 HTML 的<h1>和</h1>等相关语句。

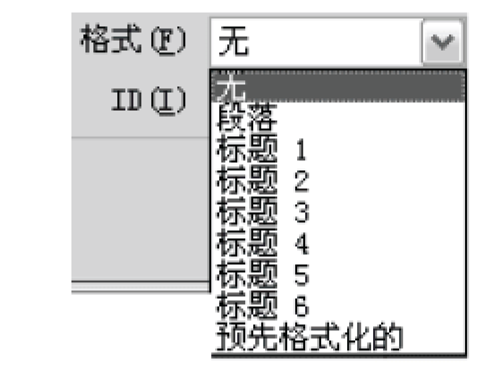


图 4-51　“格式”下拉列表框

2．设置文本的字体

在 CSS 方式属性面板的“字体”下拉列表框中可以预设部分字体，要想使用其他字体必须重新编辑字体列表，把所需的字体添加到下拉列表框中。

3．设置文本的字号

字号是指字体的大小，设置 Dreamweaver 中的字体大小有两种方法。

- 采用绝对值，在属性面板中的“大小”下拉列表框中可选择需要的字号磅数，如 9、10 等，如图 4-52 所示。
- 采用相对值，在属性面板中“大小”下拉列表框中可选择文字的相对大小，如大、较大、特大等。

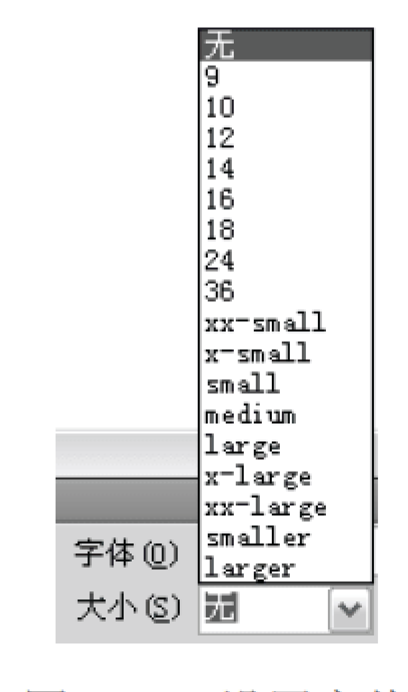


图 4-52　设置字体

4．设置文本的颜色

在“页面属性”对话框中进行的字体颜色设置，是设置整个网页所有文字的默认颜色；而选中一段文字后，在属性面板上单击“颜色”按钮，可设置选中的文字的颜色。选择“格式”→“颜色”命令，将弹出“颜色”对话框，如图 4-53 所示，

在其中也可以设置文本颜色。

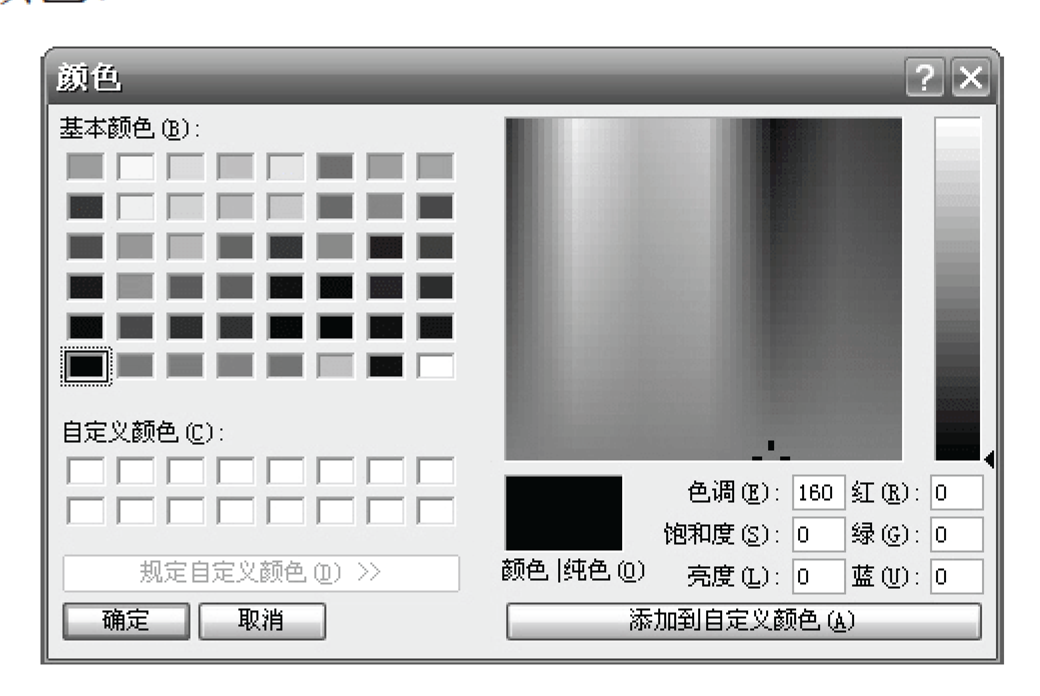


图 4-53　“颜色”对话框

4.4.3　输入特殊字符

要向网页中插入特殊字符，需要在插入栏选择“文本”类别，然后单击文本类别最右侧的按钮，弹出如图 4-54 所示的菜单，可以在其中选择所需的特殊符号插入到网页中；也可以选择“插入”→HTML→“特殊字符”命令，在打开的子菜单中选择所需要的特殊字符。

如果“特殊字符”菜单中没有需要的特殊字符，那么就选择“其他字符”命令，将弹出“插入其他字符”对话框，如图 4-55 所示，在其中单击需要的特殊字符，然后单击“确定”按钮即可将字符插入到页面的指定位置。



图 4-54　特殊字符下拉菜单



图 4-55　“插入其他字符”对话框

4.5　图 像 处 理

图像是网页最主要的表达元素之一，在网页中适当地插入图像，不但可以美化网页，还可以增强其表现力。网页对所使用的图像格式是有限制的，并不是任何一种格式的图像文件都能被应用到网页中，目前网络支持的图像格式主要有 JPEG、GIF 和 PNG 3 种。



4.5.1　插入图像

设计和规划好网页布局之后，根据设计要求，在图像处理软件中将需要插入的图片进行处理，然后存放在站点根目录下的图片文件夹中。

根据图像在网页中作用的不同，可以将其分为普通图像、占位图像和鼠标经过图像 3 种，下面分别介绍它们的插入方法。

1．插入普通图像

插入普通图像的方法如下：

（1）将鼠标指针置于需要插入图像的位置。

（2）选择“插入”→“图像”命令，弹出“选择图像源文件”对话框，如图 4-56 所示，在其中选择合适的图像，单击“确定”按钮即可。



图 4-56　“选择图像源文件”对话框

提示：除了使用“插入”菜单以外，还可以使用“插入”栏来插入图像。

2．插入占位图像

占位图像实际上是一种图像占位符，在实际制作网页的过程中，当网页的整体排版已经完成，但是需要插入的图像还没有制作出来时，可以插入一个占位图像来配合排版的需要，等到图像制作出来后，再将占位的图像替换掉。插入占位图像的方法如下：

（1）将鼠标指针置于需要插入占位图像的位置。

（2）选择“插入”→“图像对象”→“图像占位符”命令，弹出“图像占位符”对话框，如图 4-57 所示。



图 4-57　“图像占位符”对话框

（3）在“名称”文本框中输入占位图像的名称。
（4）在“宽度”和“高度”文本框中输入占位图像的宽度和高度。
（5）单击“颜色”按钮，在弹出的“颜色”面板中选择占位图像的颜色。
（6）在“替换文本”文本框中输入占位图像的替换文本。
（7）单击“确定”按钮即可插入占位图像，如图 4-58 所示。
在发布站点之前，应该使用普通图像替换所有的占位图像，方法如下：
（1）双击占位图像。
（2）在弹出的对话框中选择要替换占位图像的普通图像。
（3）单击“确定”按钮，即可将占位图像替换掉，如图 4-59 所示。

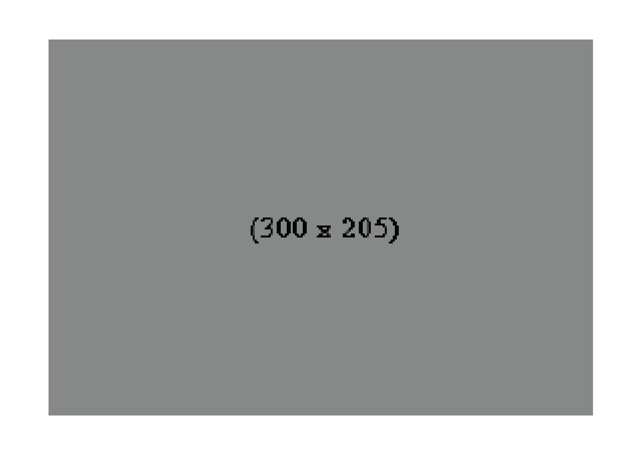


图 4-58 插入的占位图像



图 4-59 替换占位图像

3. 插入鼠标经过图像

鼠标经过图像是一种在浏览器中查看且使用鼠标指针经过时发生变化的图像，这种图像由主图像和次图像组成，如图 4-60 所示。



主图像　　次图像

图 4-60 鼠标经过图像

主图像和次图像应大小相同，如果不同，Dreamweaver CS4 将自动调整次图像的大小以匹配主图像。

在网页中插入鼠标经过图像的方法如下：
（1）将鼠标指针置于需要插入鼠标经过图像的位置。
（2）选择“插入”→“图像对象”→“鼠标经过图像”命令，弹出“插入鼠标经过图像”对话框，如图 4-61 所示。



图 4-61 “插入鼠标经过图像”对话框



（3）在“图像名称”文本框中输入鼠标经过图像的名称。

（4）在“原始图像”文本框中输入主图像所在的路径，也可以单击其右侧的按钮，在弹出的相应对话框中进行选择。

（5）在“鼠标经过图像”文本框中输入次图像所在的路径，也可以单击其右侧的按钮，在弹出的相应对话框中进行选择。

（6）在“替换文本”文本框中输入替换文本。

（7）在“按下时，前往的 URL”文本框中输入单击该图像时所链接到的文件的路径。

（8）单击“确定”按钮，完成鼠标经过图像的插入。

（9）按 F12 键，在浏览器中预览效果。

4.5.2　设置图像属性

插入图像后可根据网页的需要对其进行编辑操作，包括更改图像的大小、为图像添加边框、裁剪图像、调整图像的亮度和对比度以及锐化图像等。

1．更改图像的大小

对图像大小进行更改的操作步骤如下：

（1）选中图像，在其底部、右侧及右下角将出现控制点。

（2）然后将鼠标指针置于控制点上，任意拖动鼠标就可以更改图像的大小。

（3）可以在属性面板的“高”、“宽”文本框中输入数值直接更改图像的大小，如图 4-62 所示。



图 4-62　通过属性面板更改图像的大小

2．为图像添加边框

用户可以为网页中的图像添加边框，以达到美化图像的目的。为图像添加边框的操作步骤如下：

（1）选中要添加边框的图像。

（2）在属性面板的“边框”文本框中输入图像边框的宽度值。

（3）按 F12 键，在浏览器中预览效果，如图 4-63 所示。



图 4-63　添加边框的图像

3．裁剪图像

当在网页中插入的图像不合适时，可以使用 Dreamweaver CS4 提供的裁剪图像功能对其进行裁剪，操作步骤如下：

（1）选中要裁剪的图像。

（2）单击属性面板中的“裁剪”按钮，在其四周将出现一个裁剪控制框，如图 4-64 所示。

（3）将鼠标指针指向控制点上拖动鼠标可以改变裁剪框的大小，将鼠标指针移到裁剪框中拖动鼠标可以移动裁剪框的位置。

（4）在裁剪框中双击鼠标，可以剪切掉裁剪框以外的图像，如图 4-65 所示。



图 4-64　图像裁剪框



图 4-65　裁剪图像

4．调整图像的亮度和对比度

在 Dreamweaver 中，可以像在 Photoshop、Fireworks 等图像处理软件中一样方便地调整图像的亮度和对比度，操作步骤如下：

（1）选中要调整亮度和对比度的图像。

（2）单击属性面板中的“亮度和对比度”按钮，弹出“亮度/对比度”对话框，如图 4-66 所示。

（3）在其中设置亮度和对比度，单击“确定”按钮，即可改变图像的亮度和对比度。

5．锐化图像

通过扫描得到的图像，其边缘常常是模糊的，用户可以通过 Dreamweaver 提供的锐化图像功能提高图像的质量。锐化图像的操作步骤如下：

（1）选中要锐化的图像。

（2）单击属性面板中的“锐化”按钮，弹出“锐化”对话框，如图 4-67 所示。

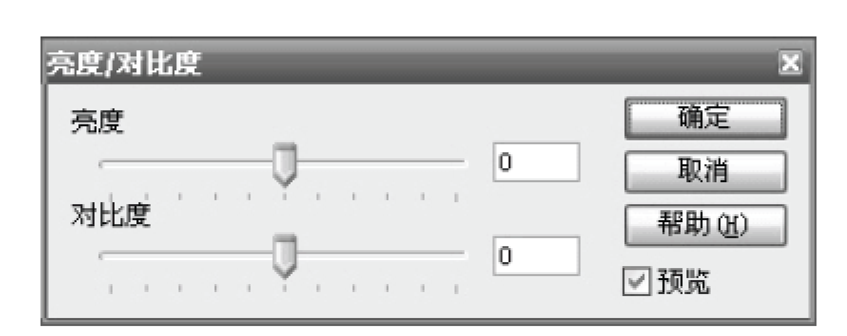


图 4-66　“亮度/对比度”对话框



图 4-67　“锐化”对话框

（3）调整的值越大，锐化程度就越大。

（4）单击“确定”按钮，结束锐化操作。

4.5.3　创建图像地图

图像地图就是在一幅图像中创建多个链接区域，单击链接区域可以跳转到链接对象中，



这种链接区域也称为热点。选择需要创建链接区域的图像，打开图片属性面板，其中的热点工具如图 4-68 所示。

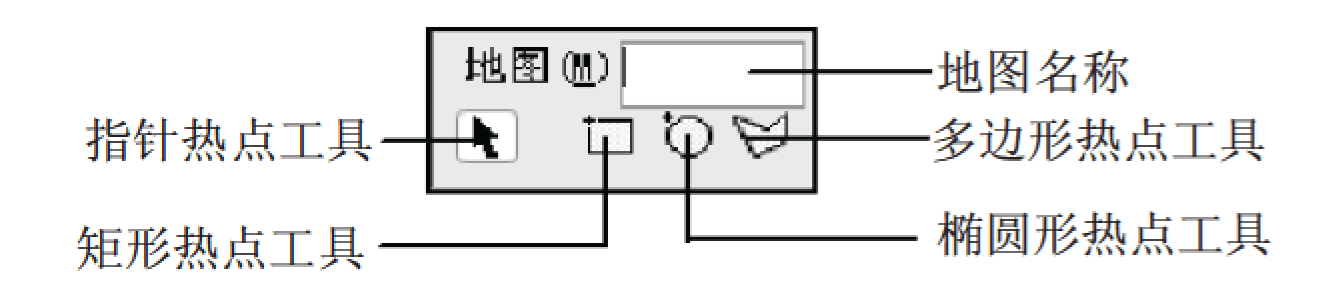


图 4-68　热点工具

在“地图”文本框中可以设置图像地图的名称。可以创建矩形热点、椭圆热点和多边形热点，如图 4-69 所示。



图 4-69　创建图像地图

使用指针热点工具选中图像热点后，在属性面板中将显示热点的相关属性，可选择热点的链接文件、链接网页的打开方式以及热点的替代信息，如图 4-70 所示。



图 4-70　热点属性面板

4.6　建立网页链接

超链接是指从一个网页指向一个目标的连接关系，这个目标可以是另一个网页，也可以是同一网页上的不同位置，还可以是一张图片、一个电子邮件地址甚至一个应用程序。

1. 链接的类型

如果按链接目标分类，可以将超链接分为内部超链接、外部超链接、锚点超链接和电子邮件超链接。

- 内部超链接：同一网站文档之间的链接。
- 外部超链接：不同网站文档之间的链接。
- 锚点超链接：同一网页或不同网页中指定位置的链接。

在网页中超链接一般分为文字超链接、锚点超链接、电子邮件超链接和图像热点超链接等几种类型。

2. 链接路径

了解从作为链接起点的文档到作为链接目标的文档之间的文件路径对于创建链接至关

重要。每个网页都有一个唯一的地址，称作统一资源定位器（URL）。不过，当创建本地链接（即从一个文档到同一站点上另一个文档的链接）时，通常不指定要链接到的文档的完整URL，而是指定一个从当前文档或站点根文件夹的相对路径。有以下 3 种类型的链接路径。

（1）绝对路径

绝对路径为文件提供完全的路径，包括适用的协议，如 http://www.haut.edu.cn。

（2）相对路径

相对路径最适合网站的内部链接。如果链接到同一目录下，则只需要输入要链接文件的名称。要链接到下一级目录中的文件，只需要输入目录名，然后输入“/”，再输入文件名。如链接到上一级目录中的文件，则先输入“../”，再输入目录名、文件名。

文档相对路径对于大多数 Web 站点的本地链接来说，是最适用的路径。在当前文档与所链接的文档处于同一文件夹内，而且可能保持这种状态的情况下，文档相对路径特别有用。文档相对路径还可用来链接到其他文件夹中的文档，方法是利用文件夹层次结构，指定从当前文档到所链接的文档的路径。

文档相对路径的基本思想是省略对于当前文档和所链接的文档都相同的绝对 URL 部分，而只提供不同的路径部分。

（3）根路径

根路径是指从站点根文件夹到被链接文档经由的路径，以前斜杠开头，例如，/zhuren/zhuren.html 就是站点根文件夹下的 zhuren 子文件夹中的一个文件（zhuren.html）的根路径，如图 4-71 所示。

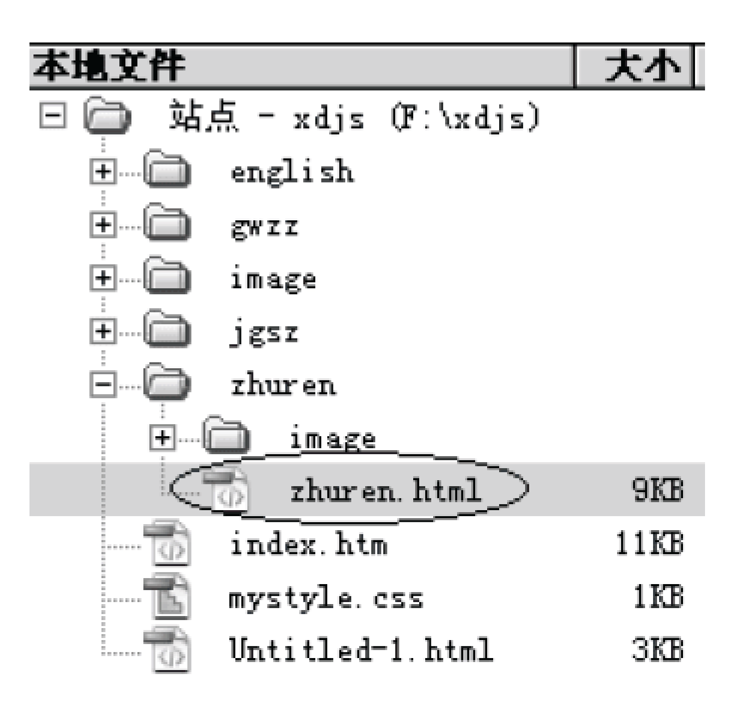


图 4-71　根文件夹

4.6.1　创建文字超链接

为文本添加超链接的操作步骤如下：

（1）选中要创建超链接的文本，如“现代技术教学部”。

（2）在属性面板的“链接”下拉列表框中输入要链接网页的地址，如图 4-72 所示。

（3）在“目标”下拉列表框中设置链接网页在浏览器中的打开方式。

（4）按 F12 键测试超链接的效果。





图 4-72　创建文字超链接

4.6.2　创建图像超链接

为图像添加超链接的操作步骤如下：

（1）选中要创建超链接的图像。

（2）在属性面板的“链接”文本框中输入要链接网页的地址，如图 4-73 所示。



图 4-73　创建图像超链接



（3）在“目标”下拉列表框中设置链接网页在浏览器中的打开方式。

（4）按 F12 键测试超链接效果。

4.6.3 创建锚点超链接

当一个网页的内容过多而无法一屏显示时，为了方便访问者浏览，需要在同一网页内部建立超链接，即创建锚点超链接。创建锚点超链接的操作步骤如下：

（1）将鼠标指针定位到要创建锚点链接的位置。

（2）单击“插入”栏的“常用”选项卡中的“命名锚记”按钮，如图 4-74 所示，弹出“命名锚记”对话框，如图 4-75 所示。



图 4-74 “命名锚记”按钮



图 4-75 “命名锚记”对话框

（3）在“锚记名称”文本框中输入锚记的名称，如“jszx”。

（4）单击“确定”按钮，则在页面中插入了锚记，如图 4-76 所示。

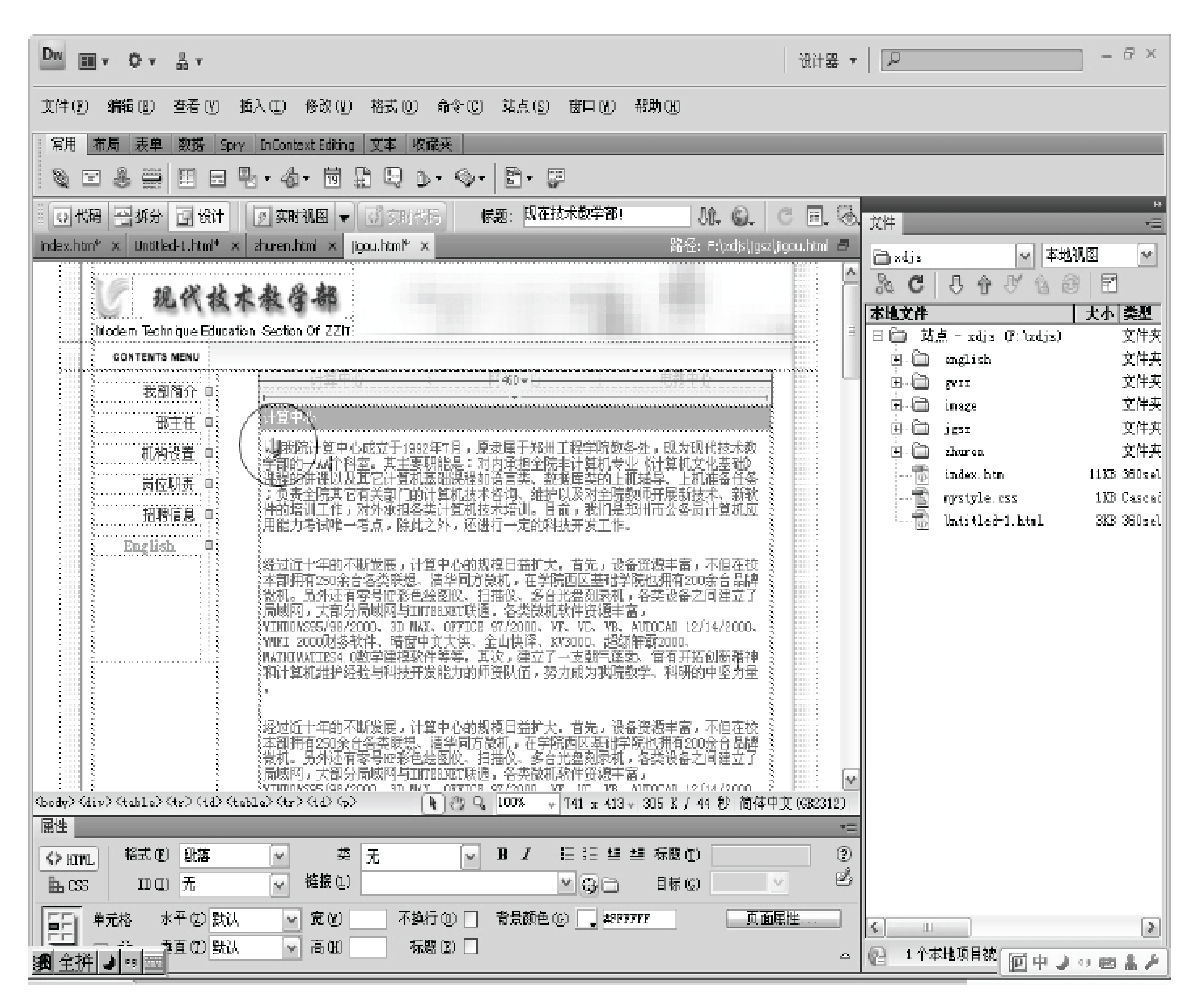


图 4-76 插入锚记

（5）在页面中选择要设置锚点超链接的文本。

（6）拖动属性面板中的“指向文件”图标到所要链接的锚记上，则在“链接”下拉列表框中将出现锚记名称，并在其之前添加“#”号，如图 4-77 所示。

（7）按 F12 键测试锚点超链接的效果。





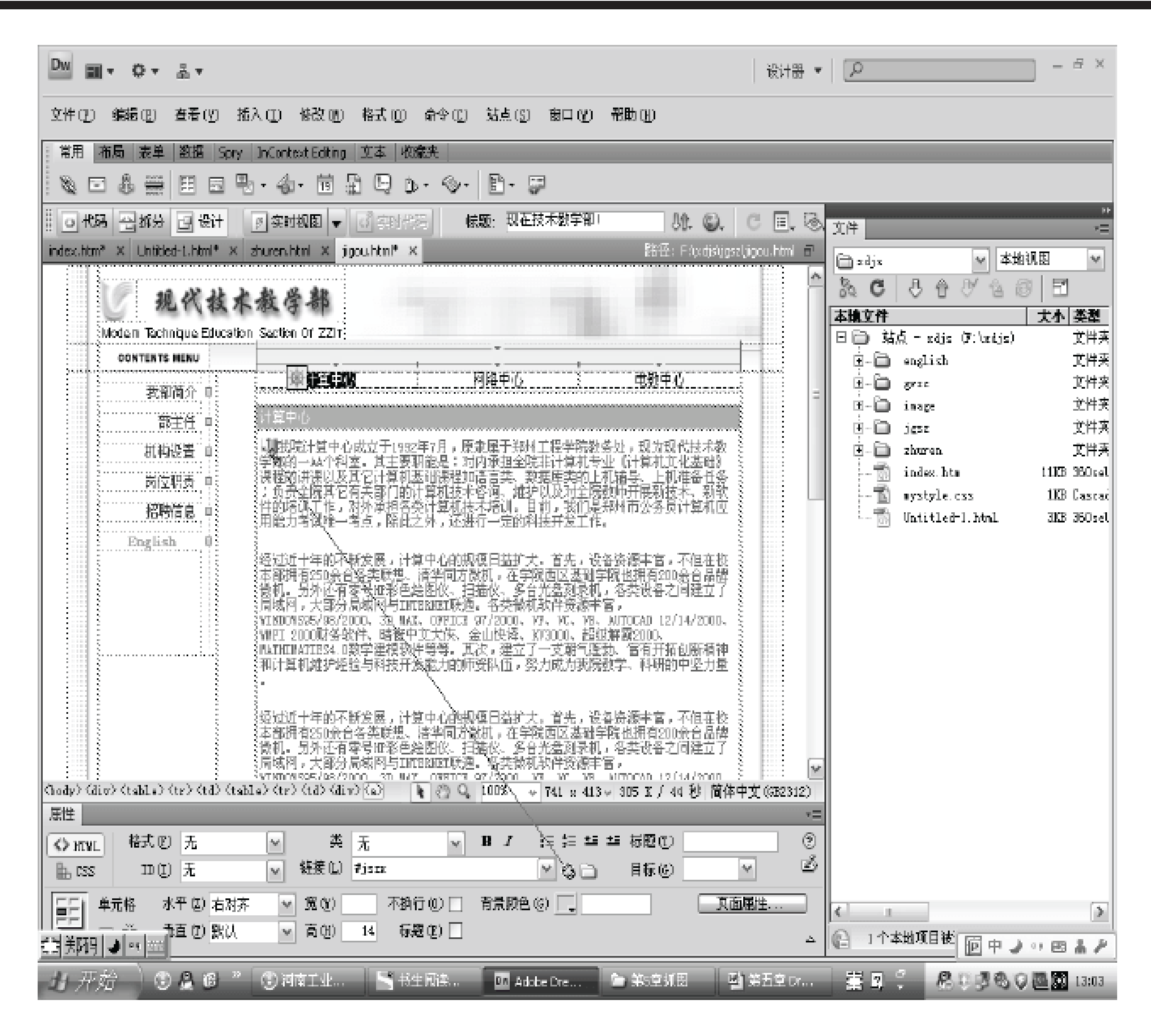


图 4-77　指向锚记

4.6.4　创建电子邮件超链接

将插入点放在希望出现电子邮件超链接的位置，或者选中作为电子邮件超链接出现的文本或图像，选择“插入”→“电子邮件链接”命令（或者在“插入”栏的“常用”选项卡中单击“电子邮件链接”按钮），弹出“电子邮件链接”对话框，如图 4-78 所示。在“文本”文本框中输入创建电子邮件超链接的文字，如“计算中心”，在 E-Mail 文本框中输入邮件地址，如“jszx@haut.edu.cn”，最后单击“确定”按钮即可。

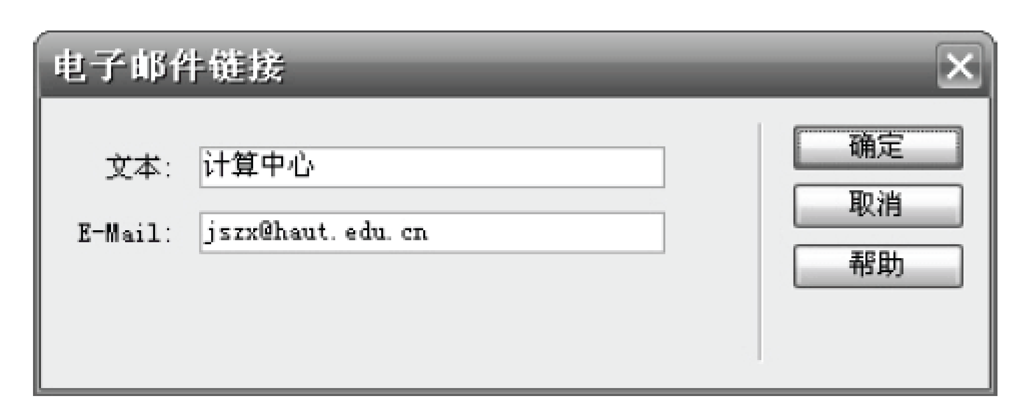


图 4-78　“电子邮件链接”对话框

电子邮件超链接创建完成后，选中该超链接，可在其属性面板中对其进行编辑。“链接”文本框中内容为“mailto: jszx@haut.edu.cn”。

4.7　表 格 处 理

表格是网页设计制作不可缺少的元素，它以简洁明了和高效快捷的方式将图片、文本、数据和表单的元素有序地显示在页面上，让用户可以设计出漂亮的页面。虽然一般用户在

浏览网页时并没有看到什么表格，而实际上绝大多数的网页都是采用无边框的表格来实现的（如图 4-79 所示）。使用表格排版的页面在不同平台、不同分辨率的浏览器中都能保持其原有的布局，而在不同的浏览器平台有较好的兼容性，所以表格是网页中最常用的排版方式之一。



图 4-79　无边框表格布局网页

4.7.1　插入表格

定位插入点，然后单击“常用”选项卡中的“表格”按钮，或者选择“插入”→“表格”命令，弹出“表格”对话框，如图 4-80 所示。设置好所需参数后，单击“确定”按钮即可插入一个表格。



图 4-80　“表格”对话框

“表格”对话框中各参数的含义如下。

- 行数：设置表格的行数。



- 列数：设置表格的列数。
- 表格宽度：设置表格的宽度，可以填入数值，紧随其后的下拉列表框用来设置宽度的单位，有两个选项，分别为百分比和像素。当宽度的单位选择百分比时，表格的宽度会随浏览器窗口大小的改变而改变。
- 边框粗细：设置表格边框的宽度，以像素为单位，默认为 1。数值越大，边框越粗。
- 单元格边距：设置单元格的边框与单元格中内容之间空白的大小。
- 单元格间距：设置单元格与单元格之间的间隔大小。
- 标题：用于设置表格的标题，有如下 4 个选项。
  - ☑ 无：对表格不启用列或行标题。
  - ☑ 左：可以将表格的第一列作为标题列，以便可为表格中的每一行输入一个标题。
  - ☑ 顶部：可以将表格的第一行作为标题行，以便可为表格中的每一列输入一个标题。
  - ☑ 两者：能够在表格中输入列标题和行标题。
- 辅助功能标题：提供一个显示在表格外的表格标题。
- 辅助功能摘要：给出表格的说明。屏幕阅读器可以读取摘要文本，但是该文本不会显示在用户的浏览器中。

表格创建好后就可在表格中添加内容。在 Dreamweaver 中可向单元格内直接输入文字、插入图像，或者将其他文档中的文字粘贴到单元格中。在表格中按 Tab 键，鼠标光标可移动到下一个单元格；按 Shift+Tab 组合键，则鼠标光标可移动到上一个单元格。

4.7.2　表格的基本操作

1. 选中表格

要对表格进行设置，首先必须先选中表格，要选择整个表格，可执行以下操作之一。

- 单击表格边框的任意处，当整个表格出现编辑点，即选中整个表格。
- 在表格内任意处单击，然后在标签选择器中单击<table>标签。
- 在单元格任意处单击鼠标右键，将弹出的快捷菜单中选择“表格”→“选择表格”命令。

2. 设置表格属性

当在“设计”视图中对表格进行格式设置时，可以设置整个表格或表格中所选行、列或单元格的属性。如果将整个表格的某个属性（如背景颜色或对齐）设置为一个值，而将单个单元格的属性设置为另一个值，则单元格格式设置优先于行格式设置，行格式设置又优先于表格格式设置。

表格格式设置的优先级别由高到低的顺序为：单元格→行数→表格。

直接插入的表格不一定能满足用户的需要，通常需要进一步进行设置。选中表格后可

以在属性面板对其进行设置，如图 4-81 所示。

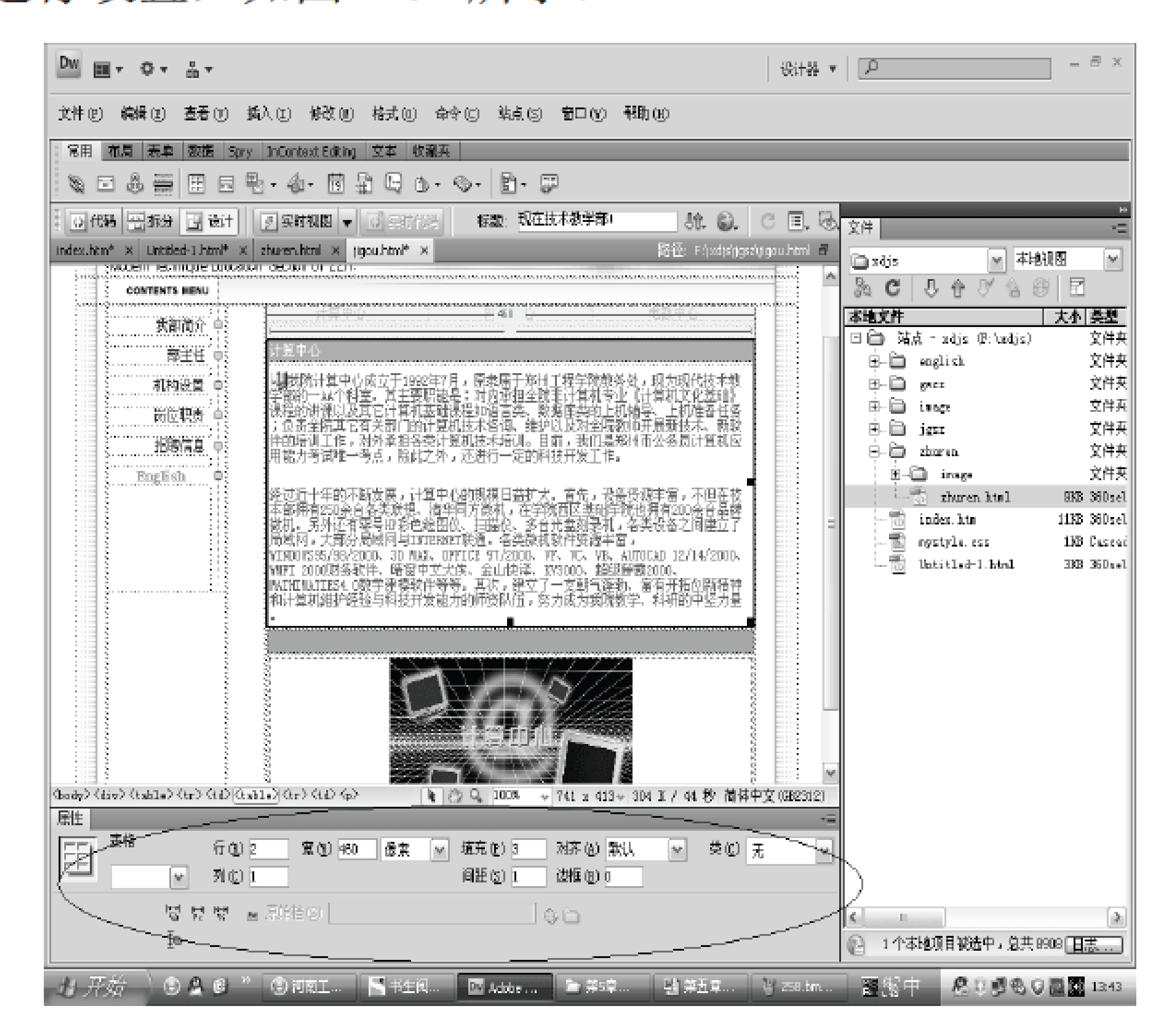


图 4-81　表格的属性面板

表格的属性面板中各参数的含义如下。

- 表格 ID：设置表格的 ID，也就是表格的名称。
- 行和列：设置表格中行和列的数目。
- 宽：设置表格的宽。是以像素为单位或按占浏览器窗口宽度的百分比计算的表格宽度和高度。
- 填充：设置单元格内容和单元格边框之间的像素数。
- 间距：设置相邻的单元格之间的像素数。
- 对齐：确定表格相对于同一段落中其他元素（如文本或图像）的显示位置，可选择左对齐、居中对齐或者右对齐。
- 边框：设置表格的边框宽度，以像素为单位。

注意：如果没有明确指定边框、单元格间距和单元格边距的值，则大多数浏览器按边框和单元格边距均设置为 1 且单元格间距设置为 2 显示表格。若要确保浏览器不显示表格中的边距和间距，可以将边框、单元格边距和单元格间距都设置为 0。

- 类：对该表格设置一个 CSS 类。
- 清除列宽和清除行高：从表格中删除所有明确指定的行高或列宽。
- 将表格宽度转换成像素和将表格高度转换成像素：将表格中每列的宽度或高度设置为以像素为单位的当前宽度（还将整个表格的宽度设置为以像素为单位的当前宽度）。
- 将表格宽度转换成百分比和将表格高度转换成百分比：将表格中每个列的宽度或高度设置为按占文档窗口宽度的百分比表示的当前宽度（还将整个表格的宽度设置为按占文档窗口宽度的百分比表示的当前宽度）。



4.7.3　行/列/单元格的基本操作

只要选中表格中的某个单元格，就可以在属性面板中设置单元格的属性。单元格属性面板可分为两部分内容，如图 4-82 和图 4-83 所示。上半部分用于设置单元格中文本的属性，类似文本的属性面板，下半部分用于设置单元格的属性。

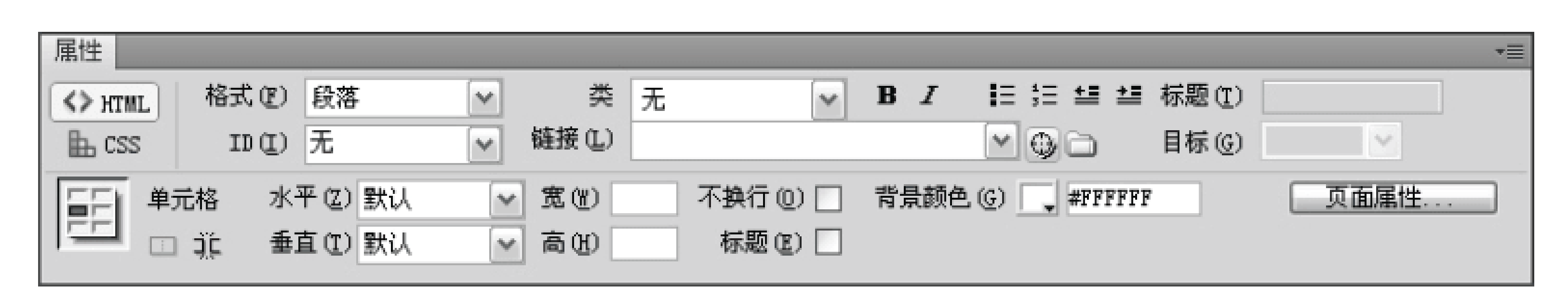


图 4-82　单元格的属性面板（HTML 方式）



图 4-83　单元格的属性面板（CSS 方式）

单元格属性面板中各参数的含义如下。

- 水平：指定单元格、行或列内容的水平对齐方式。可以将内容对齐到单元格的左侧、右侧或使其居中对齐，也可以指示浏览器使用其默认的对齐方式（通常常规单元格为左对齐，标题单元格为居中对齐）。
- 垂直：指定单元格、行或列内容的垂直对齐方式。可以将内容对齐到单元格的顶端、中间、底部或基线，或者指示浏览器使用其默认的对齐方式（通常是中间）。
- 宽和高：所选单元格的宽度和高度，以像素为单位或按整个表格宽度或高度的百分比指定。若要指定百分比，则在值后面使用百分比符号（%）。若要让浏览器根据单元格的内容以及其他列和行的宽度和高度确定适当的宽度或高度，可将此区域留空（默认设置）。默认情况下，浏览器选择行高和列宽的依据是能够在列中容纳最宽的图像或最长的行。

注意：可以按占表格总高度的百分比指定一个高度，但是浏览器中的行可能不以指定的百分比高度显示。

- 背景颜色：使用颜色选择器选择的单元格、列或行的背景颜色。
- 边框：单元格的边框颜色（“页面属性”按钮）。
- 合并单元格：将所选的单元格、行或列合并为一个单元格。只有当单元格形成矩形或直线的块时才可以合并这些单元（“页面属性”按钮）。
- 拆分单元格：将一个单元格分成两个或更多个单元格。一次只能拆分一个单元格；如果选择的单元格多于一个，则此按钮将禁用（“页面属性”按钮）。
- 不换行：防止换行，从而使给定单元格中的所有文本都在一行上。如果选中“不换行”复选框，则当输入数据或将数据粘贴到单元格时单元格会加宽来容纳所有

数据（通常，单元格在水平方向扩展以容纳单元格中最长的单词或最宽的图像，然后根据需要在垂直方向进行扩展以容纳其他内容）。

- 标题：将所选的单元格格式设置为表格标题单元格。默认情况下，表格标题单元格的内容为粗体并且居中。

4.7.4 表格的高级操作

1. 选中行

要选择某一行或某一列，将鼠标光标移动到行左侧或列上方，鼠标指针变为向右或向下的箭头图标时单击，就可选中整行或整列。另外，在标签选择器中单击<tr>标签，就可以选择一整行。

2. 选中列

要选中某一单元格，可单击需要选中的单元格即可，或者在标签选择器中单击<td>标签。要选中连续的单元格，按住鼠标左键从一个单元格的左上方开始向连续选择单元格的方向拖动即可，要选中不连续的几个单元格，可以按住 Ctrl 键，连续单击要选择的所有单元格。另外，先单击一个单元格，然后按住 Shift 键再单击另一个单元格，则由这两个单元格组成的矩形区域里的所有单元格被选中。

3. 插入行或列

若要在所需位置插入一行（或一列），则将鼠标光标定位在需要插入行（或列）的单元格内，然后选择“插入”→“表格对象”→“在上面插入行”（或“在下面插入行”、“在左边插入列”、“在右边插入列”）命令，可插入指定的行或列。

将鼠标光标定位在表格中，选择“修改”→“表格”→“插入行”命令，会在当前行的上面插入一行。

将鼠标光标定位在表格中，选择“修改”→“表格”→“插入列”命令，会在当前列的左侧插入一列。

如果要添加多行（列），则将光标定位在某一单元格中，右击，在弹出的快捷菜单中选择“表格”→“插入行或列”命令，弹出如图 4-84 所示的“插入行或列”对话框，设置好所有内容，单击“确定”按钮完成插入。

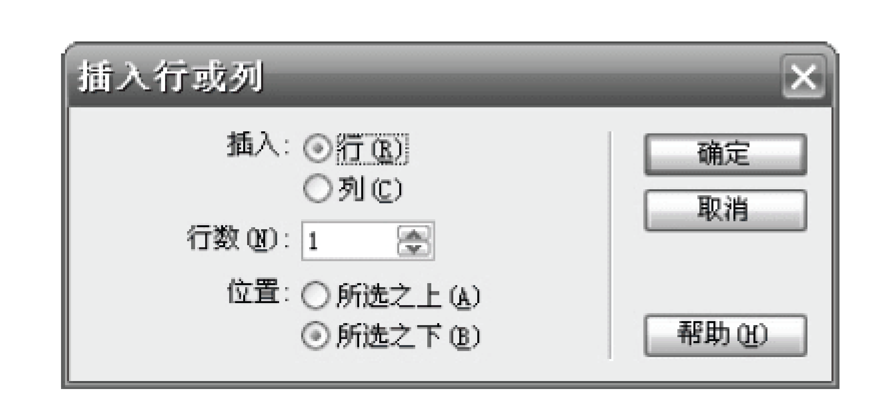


图 4-84 “插入行或列”对话框

4. 删除行、列

选中整行或整列后，按 Delete 键即可删除，或者选中整行（或整列），单击鼠标右键，在弹出的快捷菜单中选择“表格”→“删除行”（或“删除列”）命令即可删除该行或列表格。要删除整张表格先要选中表格，然后按 Delete 键删除。



5．拆分单元格

选中要拆分的单元格，然后选择“修改”→“表格”→“拆分单元格”命令，弹出如图 4-85 所示的“拆分单元格”对话框，可选择拆分为行或者列，并输入行数或者列数，单击“确定”按钮，即可完成对单元格的拆分。同样也可以在选中单元格的状态下，通过表格属性面板中的“拆分”按钮来拆分单元格。



图 4-85　“拆分单元格”对话框

6．合并单元格

选中行或列中连续的单元格，然后选择“修改”→“表格”→“合并单元格”命令，多个单元格就合并为一个单元格。合并前各单元格的内容将放置在合并后的单元格中，所选的第一个单元格的属性将应用于合并的单元格。也可以在选中单元格的状态下，通过表格属性面板中的“合并”按钮来合并单元格。

4.8　CSS 样式表

CSS 样式（层叠样式表）是一系列格式设置规则，它们控制 Web 页面内容的外观。使用 CSS 设置页面格式时，是将内容与表现形式分开。页面内容（即 HTML 代码）驻留在 HTML 文件自身中，而用于定义代码表现形式的 CSS 规则驻留在另一个文件（外部样式表）或 HTML 文档的另一部分（通常为文件头部分）中。使用 CSS 可以非常灵活并更好地控制具体的页面外观（从精确的布局定位到特定的字体和样式）。

CSS 格式设置规则由两部分组成，即选择器和声明。选择器是标识格式元素的术语（如 p、h1、类名或 id），声明用于定义元素样式。在下面的示例中，h1 是选择器，大括号之间的所有内容都是声明：

```
p {
 FONT-FAMILY: "宋体";
FONT-SIZE: 14px;
}
```

声明由两部分组成，分别为属性（如 FONT-FAMILY）和值（如"宋体"）。上面的 CSS 规则为 p 标签创建了一个特定的样式：链接到此样式的所有 p 标签的文本都将是 14 个像素大小、宋体。

4.8.1　创建 CSS 样式表

创建新的 CSS 样式表的步骤如下：

（1）选择“窗口”→“CSS 样式”命令，打开“CSS 样式”面板，如图 4-86 所示，单击右下角的“新建 CSS 规则”按钮，打开“新建 CSS 规则”对话框，如图 4-87 所示。



图 4-86 “CSS 样式”面板



图 4-87 “新建 CSS 规则”对话框

（2）在“新建 CSS 规则”对话框中，指定要创建的 CSS 规则的选择器类型。

- 若要创建一个可作为 class 属性应用于任何 HTML 元素的自定义样式，从“选择器类型”下拉列表框中选择“类”选项，然后在“选择器名称”文本框中输入样式的名称。
- 若要定义包含特定 ID 属性的标签的格式，可从“选择器类型”下拉列表框中选择 ID 选项，然后在“选择器名称”文本框中输入唯一 ID。
- 若要重新定义特定 HTML 标签的默认格式，可从“选择器类型”下拉列表框中选择“标签”选项，然后在“选择器名称”文本框中输入 HTML 标签或从下拉列表框中选择一个标签。
- 若要定义同时影响两个或多个标签、类或 ID 的复合规则，则选择“复合内容”选项并输入用于复合规则的选择器。例如，如果输入“div p”，则 div 标签内的所有 p 元素都将受此规则影响。在中间的列表框中会准确说明添加或删除选择器时该规则将影响哪些元素，如图 4-88 所示。

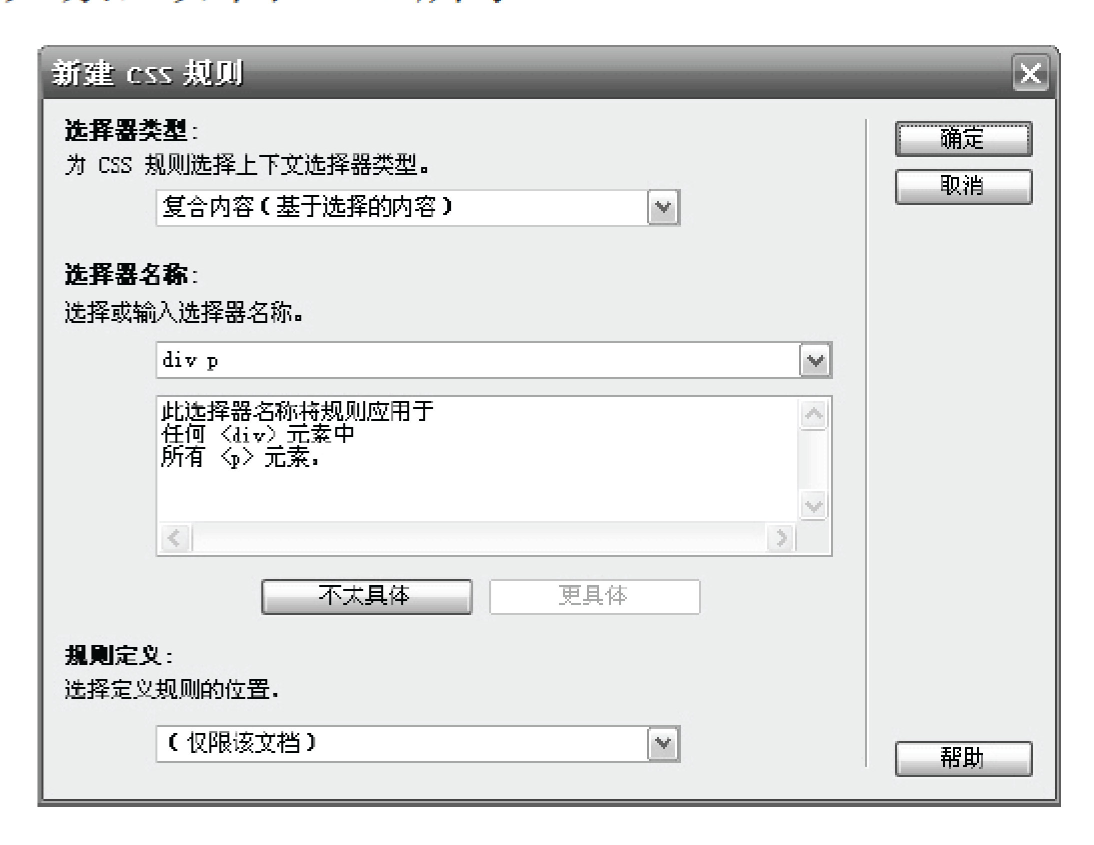


图 4-88 “复合内容”选项

（3）选择要定义规则的位置，然后单击“确定”按钮。



- 若要将规则放置到已附加到文档的样式表中，则选择相应的样式表。
- 若要创建外部样式表，则选择“新建样式表文件”选项。
- 若要在当前文档中嵌入样式，则选择“仅对该文档”选项。

（4）在“CSS 规则定义”对话框中选择要为新的 CSS 规则设置的样式选项。

（5）完成对样式属性的设置后，单击“确定”按钮。

4.8.2　CSS 属性设置

创建新的 CSS 样式表之后，可以定义 CSS 规则的属性，如文本字体、背景图像和颜色、间距和布局属性以及列表元素外观。从“CSS 样式”面板中选择需要重新设置的 CSS 样式后，单击面板右下侧的编辑样式按钮，可打开 CSS 规则定义对话框编辑 CSS 样式，如图 4-89 所示。

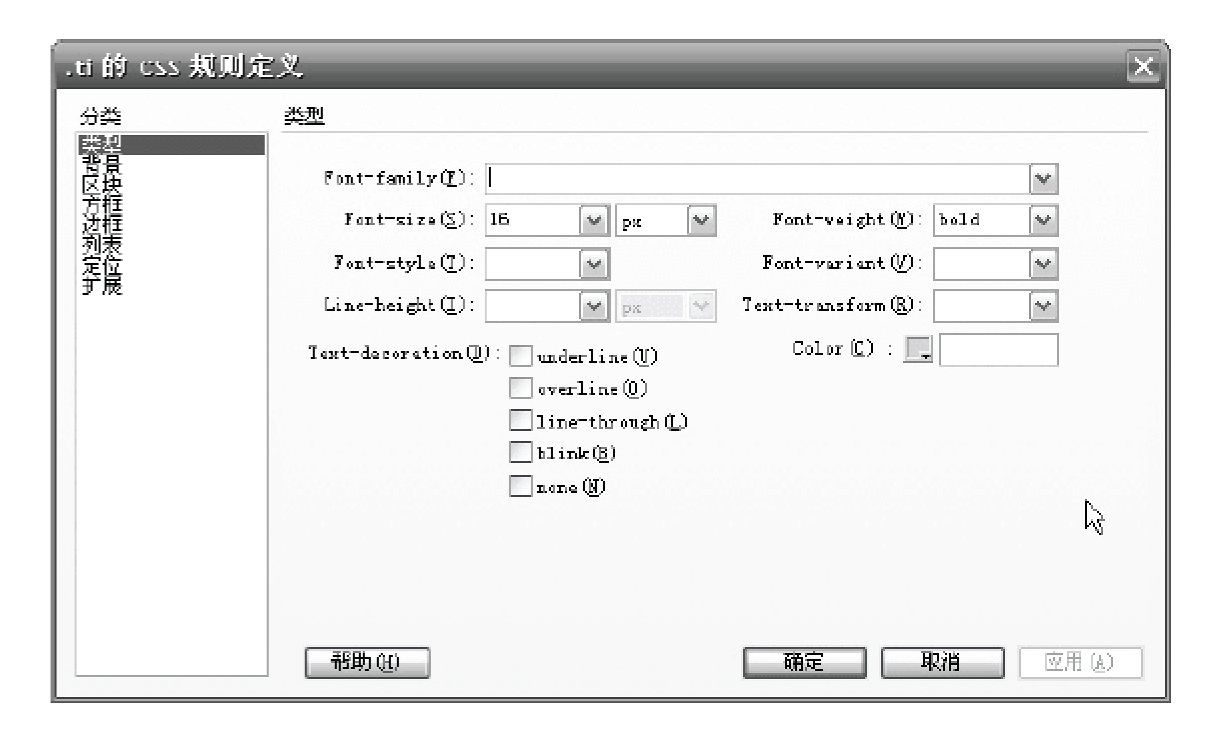


图 4-89　“.ti 的 CSS 规则定义”对话框

在“.ti 的 CSS 规则定义”对话框中可以定义 CSS 的类型、背景、区块、方框、边框、列表、定位和扩展等属性。

注意：限于篇幅，关于各 CSS 样式属性的详细说明，此处不再做详细说明，读者可参考 Dreamweaver CS4 的帮助文档。

4.8.3　应用 CSS 样式

（1）在文档中，选择要应用 CSS 样式的文本，然后将插入点放在段落中，以便将样式应用于整个段落。如果在单个段落中选择一个文本范围，则 CSS 样式只影响所选范围。若要指定要应用 CSS 样式的确切标签，可在位于文档窗口左下角的标签选择器中选择标签。

（2）若要应用类样式，可执行下列操作之一：

- 在“CSS 样式”面板（选择“窗口”→“CSS 样式”命令打开）中，选择“全部”模式，右击要应用的样式的名称，然后从弹出的快捷菜单中选择“应用”命令。
- 在 HTML 属性检查器中，从“类”下拉列表框中选择要应用的类样式。
- 在文档窗口中，右击所选文本，在弹出的快捷菜单中选择“CSS 样式”命令，然

后选择要应用的样式。

- 选择“格式”→“CSS 样式”命令，然后在子菜单中选择要应用的样式。

4.9 嵌入表单元素

表单（Form）是一种常见的页面元素，它一般用来实现服务器与用户之间的交互，有时也用于实现某些动态功能（如创建能响应用户操作的按钮）。表单是用于实现网页浏览者与服务器之间信息交互的一种页面元素，在 WWW 上它被广泛用于各种信息的搜集和反馈。例如，图 4-90 所示网页中显示了一个用于进行电子邮件系统登录的表单（图中椭圆形标记的位置），在这个表单中，包含一些简单的文字和两个文本框（一个文本框和一个密码框），另外还有一个“登录”按钮，当浏览者在文本框中填写数据后单击“登录”按钮，则填写的内容将被传送到服务器，由服务器进行具体的处理，然后确定下一步的操作。



图 4-90 表单示例

不论是什么类型的表单，它的基本工作原理都是一样的，那就是访问者浏览到表单页面后，在表单中填写或选择必要的信息，最后单击“提交”按钮（按钮名称可能是“注册”、“同意”、“登录”等），然后填写或选择的信息就按照设计者指定的方式发送到 Web 服务器端，由服务器端特定的程序进行处理，之后通常会向访问者返回一个页面，以对用户提交的信息给予回复。

处理表单数据通常是在服务器端，一般采用 ASP 程序或 PHP 等脚本程序，也可以使用 C#等编写的动态链接库程序。如果用户的需求比较简单，可以申请使用免费的表单和相应处理程序，否则就需要自行编写表单处理程序。有关自定义表单处理程序的详细信息，请参考本书的其他章节内容。

总之，表单不同于前面介绍的页面元素（如表格、图像等），它不但需要在网页中用 HTML 进行显示，而且还需要服务器端特定程序的支持。

4.9.1 表单对象

在 Dreamweaver 中，表单输入类型称为表单对象。表单对象是允许用户输入数据的机



制，可以在表单中添加以下表单对象。

（1）文本域

接受任何类型的字母、数字、文本输入内容。文本可以单行或多行显示，也可以以密码域的方式显示，在这种情况下，输入文本将被替换为星号或项目符号，可以防止文本泄露，起到保密的作用。

（2）隐藏域

存储用户输入的信息，如姓名、电子邮件地址或偏爱的查看方式，并在该用户下次访问此站点时使用这些数据。

（3）按钮

在单击时执行操作。可以为按钮添加自定义名称或标签，或者使用预定义的“提交”或“重置”标签。使用按钮可将表单数据提交到服务器或者重置表单，还可以指定其他已在脚本中定义的处理任务。例如，可能会使用按钮根据指定的值计算所选商品的总价。

（4）复选框

允许在一组选项中选择多个选项。用户可以选择任意多个适用的选项。

（5）单选按钮

代表互相排斥的选择。在某单选按钮组（由两个或多个共享同一名称的按钮组成）中选择一个按钮，就会取消选择该组中的所有其他按钮。

（6）列表菜单

在一个滚动列表中显示选项值，用户可以从该滚动列表中选择多个选项。“列表”选项在一个菜单中显示选项值，用户只能从中选择单个选项。只有有限的空间但必须显示多个内容项，或者要控制返回给服务器的值时，建议使用菜单。

菜单与文本域不同，在文本域中用户可以随心所欲地输入任何信息，甚至包括无效的数据，对于菜单而言，则可以具体设置某个菜单返回的确切值。

（7）跳转菜单

可导航的列表或弹出菜单，使用它们可以插入一个菜单，其中的每个选项都链接到某个文档或文件。

（8）文件域

使用户可以浏览到其计算机上的某个文件并将该文件作为表单数据上传。

（9）图像域

使用户可以在表单中插入一个图像。使用图像域可生成图形化按钮，如“提交”或“重置”按钮。如果使用图像来执行任务而不是提交数据，则需要将某种行为附加到表单对象。

4.9.2　创建表单

创建表单的操作步骤如下：

（1）打开一个页面，将插入点放在希望表单出现的位置。

（2）选择“插入”→“表单”命令，或选择“插入”面板中的“表单”类别，然后选

择“表单”选项。

在“设计”视图中，表单以红色的虚轮廓线指示。如果看不到这个轮廓线，可选择“查看”→“可视化助理”→“不可见元素”命令。

（3）在属性面板（选择“窗口”→“属性”命令）中设置 HTML 表单的属性，如图 4-91 所示。在“表单 ID”文本框中输入标识该表单的唯一名称，在“动作”文本框中输入路径或者单击文件夹图标导航到相应的页面或脚本，以指定将处理的表单数据的页面或脚本。在“方法”下拉列表框中指定将表单数据传输到服务器的方法，有以下 3 个选项。



图 4-91　表单的属性面板

- 默认设置：将表单数据发送到服务器。通常，默认值为 GET 方法。
- GET：将值附加到请求该页面的 URL 中。
- POST：在 HTTP 请求中嵌入表单数据。

（4）在页面中插入表单对象。

（5）调整表单的布局。

4.10　添加多媒体元素

多媒体技术是集计算机技术、音频处理技术、视频处理技术、图像处理技术于一体的综合技术，具有多样性、交互性和实时性等特点，在网页中添加多媒体元素，使网页绘声绘色且活动起来，可以给浏览者带来听觉和视觉的冲击，提高可视性。

4.10.1　插入网页背景音乐

声音能极好地烘托网页的氛围，网页中常见的声音格式有 wav、mp3、midi 等。在确定采用哪种格式和方法添加声音前，需要考虑添加声音的目的、页面访问者、文件大小、声音品质和不同浏览器的差异等因素。浏览器不同，处理声音文件的方式也会有很大差异。

1．.midi 或.mid（Musical Instrument Digital Interface，乐器数字接口）

许多浏览器都支持 MIDI 文件，并且不需要插件。尽管 MIDI 文件的声音品质非常好，但也可能因访问者的声卡配置而有一定的差异。很小的 MIDI 文件可以提供较长时间的声音剪辑。MIDI 文件不能进行录制，并且必须使用特殊的硬件和软件在计算机上合成。

2．.wav（波形扩展）

WAV 类文件具有良好的声音品质，许多浏览器都支持此类格式文件并且不需要插件。可以用 CD、磁带、麦克风等录制 WAV 文件。但是，其较大的文件大小严格限制了可以在



网页上使用的声音剪辑的长度。

3．.aif（Audio Interchange File Format，音频交换文件格式或 AIFF）

AIFF 格式与 WAV 格式类似，也具有较好的声音品质，大多数浏览器都可以播放它并且不需要插件。

4．.mp3（Motion Picture Experts Group Audio Layer-3，运动图像专家组音频第 3 层或称为 MPEG 音频第 3 层）

一种压缩格式，它可使声音文件明显缩小。其声音品质非常好：如果正确录制和压缩 MP3 文件，其音质甚至可以和 CD 相媲美。MP3 技术可以对文件进行“流式处理”，访问者不必等待整个文件下载完成即可收听。但是，其文件大小要大于 RealAudio 文件，因此通过典型的拨号（电话线）调制解调器连接下载整首歌曲可能仍要花较长的时间。若要播放 MP3 文件，访问者必须下载并安装辅助应用程序或插件，如 QuickTime、Windows Media Player 或 RealPlayer。

5．.ra、.ram、.rpm 或 Real Audio

此格式具有非常高的压缩度，文件大小要小于 MP3。全部歌曲文件可以在合理的时间范围内下载。因为可以在普通的 Web 服务器上对这些文件进行“流式处理”，所以在文件完全下载完之前就可听到声音。必须下载并安装 RealPlayer 辅助应用程序或插件才可以播放这种文件。

注意： 除了上面列出的比较常用的格式外，还有许多不同的音频和视频文件格式可在 Web 上使用。

在页面中可以添加背景音乐，背景音乐多以 MP3、MIDI 文件为主，在 Dreamweaver CS4 中，添加背景音乐有以下两种方法。

（1）通过添加代码的方式，步骤如下：

① 打开要添加背景音乐的网页，切换到代码视图或混合视图。

② 在<head>和</head>之间添加代码“<bagsound src=背景音乐的 URL>”，如图 4-92 所示。



图 4-92　添加背景音乐

（2）通过行为实现。利用 Dreamweaver CS4 提供的播放声音行为可实现背景音乐的播放，具体步骤如下：

从“行为”面板的“添加行为”菜单中选择“~ 建议不再使用”命令，弹出“播放声音”对话框，如图 4-93 所示。



图 4-93 “播放声音”对话框

单击“浏览”按钮可选择需要插入的声音文件（也可以直接输入地址），弹出“选择文件”对话框，在其中，除了可以选择所需的音乐文件外，还可以单击对话框中的“参数”按钮，弹出“参数”对话框，设置背景音乐自动开始等属性，如图 4-94 所示。最后单击“确定”按钮即可。

注意：此行为从 Dreamweaver CS4 开始已被弃用。



图 4-94 “选择文件”对话框和“参数”对话框

4.10.2 插入视频

在 Dreamweaver 文档中可以插入 QuickTime 或 Shockwave 影片、Java Applet、ActiveX 控件或其他音、视频对象。用户可以下载视频，或者可以对视频进行流式处理，以便在下载它的同时播放它。插入视频的步骤如下：

（1）将视频剪辑文件放入站点根目录的相应文件夹中，这些剪辑文件通常采用 AVI 或 MPEG 格式。

（2）链接到某个视频剪辑文件，或将其嵌入到页面中。

若要链接到视频剪辑文件，则需要输入链接文本（如“下载剪辑”），选择文本，然后在属性面板中单击文件夹图标，浏览到视频文件然后选择它。

将视频剪辑文件嵌入网页中的步骤如下：

① 将插入点放在文档窗口中希望插入对象的位置。

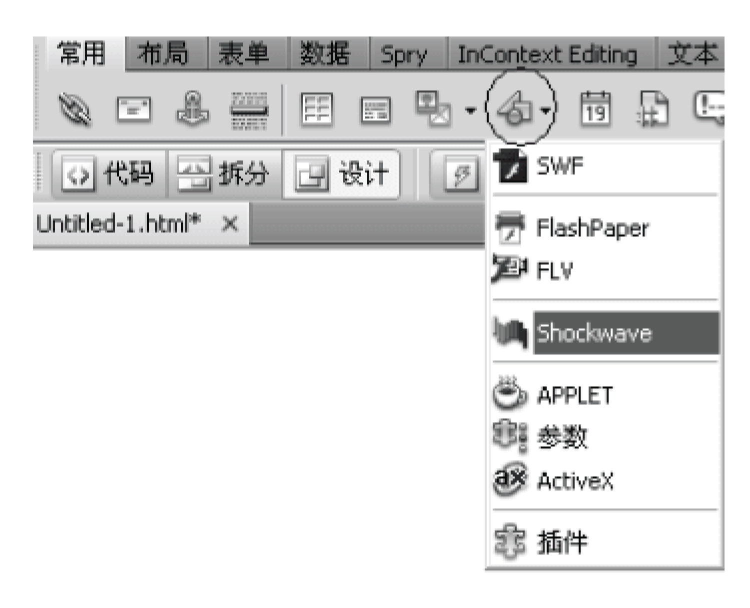
② 执行下列操作之一插入对象。

- 在“插入”面板的“常用”类别中选择“媒体”选项，并选择要插入的对象类型



的图标，如图 4-95 所示。

- 从“插入”菜单的“媒体”子菜单中选择适当的对象，如图 4-96 所示。如果要插入的对象不是 Shockwave、Applet 或 ActiveX 对象，可从“媒体”子菜单中选择“插件”命令。将打开“选择文件”对话框，可从中选择源文件并为媒体对象指定某些参数。



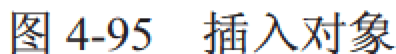

图 4-95　插入对象



图 4-96　“媒体”子菜单

③ 弹出“选择文件”对话框，然后单击“确定”按钮。

④ 单击“确定”按钮插入媒体对象。

注意：用户必须下载辅助应用程序（如插件）才能查看常见的流式处理格式，如 Real Media、QuickTime 和 Windows Media。

4.10.3　插入 Flash 动画

在 Internet 上还有这样一类的网站，它可以将音乐、声效、动画及富有新意的界面融合在一起，以达到高品质的网页动态效果，这类网站典型的代表就是 Flash 效果，它是由 Flash 动画软件制作的。

Flash 动画是一种矢量动画格式，具有体积小、兼容性好、直观动感、互动性强大、支持 MP3 音乐等优点，是当今最流行的 Web 页面动画格式。Flash 是目前最好的网络多媒体动画，它可以通过文字、图片、录像、声音等综合手段形象地体现一个意图，强烈的视觉冲击力可以给浏览者留下深刻的印象。

Flash CS4 是美国 Adobe 公司开发的矢量图形编辑和动画创作的专业软件，在使用 Dreamweaver 来插入使用 Adobe Flash 创建的内容之前，首先来了解一下不同的文件类型。

- FLA 文件（.fla）：此类型的文件只能在 Flash 中打开（而无法在 Dreamweaver 或浏览器中打开）。可以在 Flash 中打开 FLA 文件，然后将它发布为 SWF 或 SWT 文件以在浏览器中使用。
- SWF 文件（.swf）：FLA（.fla）文件的编译版本，已进行优化，可以在 Web 上查看。此文件可以在浏览器中播放并且可以在 Dreamweaver 中进行预览，但不能在 Flash 中编辑。

在网页中添加 Flash 动画的步骤如下：

（1）在文档窗口的“设计”视图中，将插入点放置在要插入内容的位置，然后执行以下操作之一：

- 在“插入”面板的“常用”类别中选择“媒体”选项，然后选项 SWF 选项，如图 4-95 所示。
- 选择“插入”→“媒体”→SWF 命令，如图 4-96 所示。

（2）在打开的对话框中，选择一个 SWF 文件（.swf），如图 4-97 所示。

（3）设置辅助功能属性，如图 4-98 所示。



图 4-97　选择 SWF 文件



图 4-98　设置辅助功能属性

（4）单击“确定”按钮之后在文档窗口中显示一个 SWF 文件占位符，如图 4-99 所示。

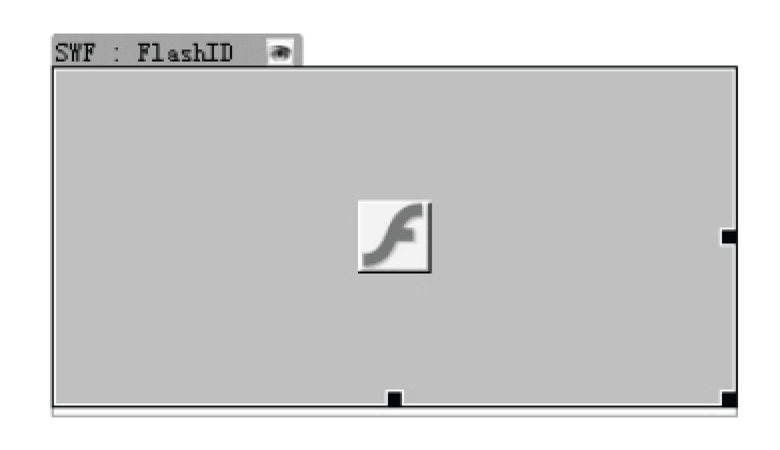


图 4-99　SWF 文件占位符

注意：此占位符有一个选项卡式的蓝色外框。此选项卡指示资源的类型（SWF 文件）和 SWF 文件的 ID，还显示一个眼睛图标，此图标可用于在 SWF 文件和用户在没有正确的 Flash Player 版本时看到的下载信息之间切换。

添加完成之后，选择一个 SWF 文件，，然后在属性面板中设置 SWF 文件属性，如图 4-100 所示。限于篇幅，关于各属性的详细配置此处不再详细介绍，读者可参阅帮助文档。



图 4-100　设置 SWF 文件属性



4.11 框架的使用

框架是网页中经常使用的页面设计方式，其作用就是把网页在一个浏览器窗口下分割成几个不同的区域，实现在一个浏览器窗口中显示多个 HTML 页面的功能。使用框架可以非常方便地完成导航工作，让网站的结构更加清晰，而各个框架之间不存在干扰问题。利用框架最大的特点就是使网站的格局一致。通常把一个网站中页面相同的部分单独制作成一个页面，作为框架结构的一个子框架的内容给整个网站公用。

一个框架结构由如下两部分网页文件构成。

- 框架（Frame）：是浏览器窗口中的一个区域，它可以显示与浏览器窗口的其余部分中所显示内容无关的网页文件。
- 框架集（Frameset）：也是一个网页文件，它将一个窗口通过行和列的方式分割成多个框架，框架的多少根据具体有多少网页来决定，每个框架中要显示的就是不同的网页文件。

4.11.1 创建框架集

在创建框架集或使用框架前，通过选择“查看”→“可视化助理”→“框架边框”命令，使框架边框在文档窗口的设计视图中可见。创建框架集有以下几种方式：

- 由新建文件创建框架集。在“新建文档”对话框中，选择“示例文件夹”列表框中的“框架页”选项，然后在“示例文件夹”列表框选择某种框架集，如图 4-101 所示。单击“创建”按钮，完成框架集的创建。



图 4-101　新建框架文档

- 由“插入”栏“布局”类别创建框架集。将“插入”栏设置为“布局”，单击“框架”按钮，会弹出下拉菜单，从中选择一种框架集也可以完成创建，如图 4-102

所示。

- 单击“创建”按钮后，会弹出“框架标签辅助功能属性”对话框，在其中可以为每个框架设置标题，如图 4-103 所示。

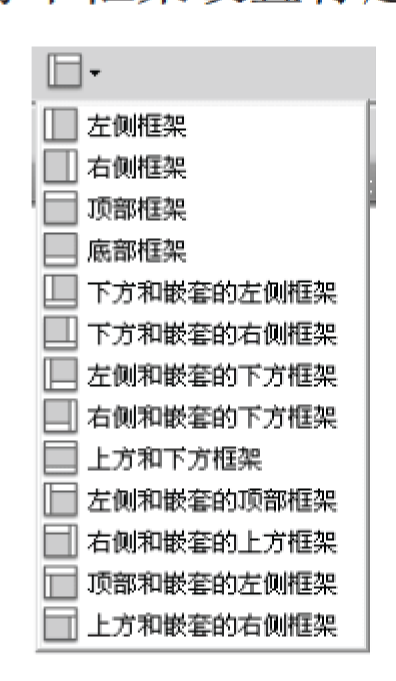




图 4-102　工具栏



图 4-103　“框架标签辅助功能属性”对话框

4.11.2　框架和框架集的基本操作

1．选择框架与框架集

选择框架与框架集的步骤如下：

（1）选择“窗口”→“框架”命令，打开“框架”面板，可以看到每一个框架都有一个框架名称（系统为其设置了默认的框架名称），如图 4-104 所示。在选中某一框架时，可以在其属性面板修改名称。

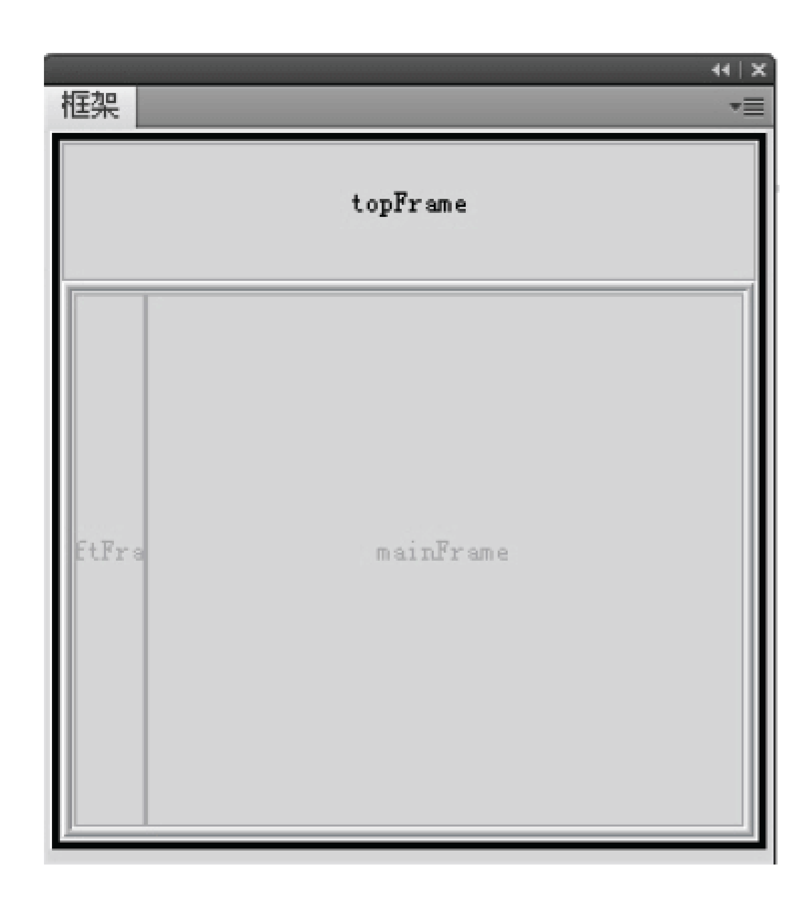


图 4-104　“框架”面板

（2）在“框架”面板中单击框架名称即可选中框架，单击较粗的边框即可选中框架集。

2．保存框架与框架集

在浏览器中预览框架集前，必须保存框架集文件以及框架集中的每个框架。可以单独保存每个框架集文件和带框架的文档，也可以同时保存框架集文件和框架中出现的所有文档。



注意：在使用 Dreamweaver 中的可视工具创建一组框架时，框架中显示的每个新文档都将获得一个默认文件名。例如，第一个框架集文件被命名为 UntitledFrameset-，而框架被命名为 UntitledFrame-。

（1）保存框架集文件

首先在“框架”面板或文档窗口中选择框架集，然后执行下拉操作即可保存框架集文件。

- 若要保存框架集文件，可选择“文件”→“保存框架集”命令。
- 若要将框架集文件另存为新文件，可选择“文件”→“框架集另存为”命令。

（2）保存框架文件

在“框架”面板或文档窗口中选择某个框架，然后选择“文件”→“保存框架”命令或选择“文件”→“框架另存为”命令即可保存框架文件。

4.11.3　设置框架属性

1. 查看或设置框架属性

（1）在“框架”面板中选择某个框架。

（2）在属性面板中，单击右下角的展开箭头，查看所有框架属性，如图 4-105 所示。

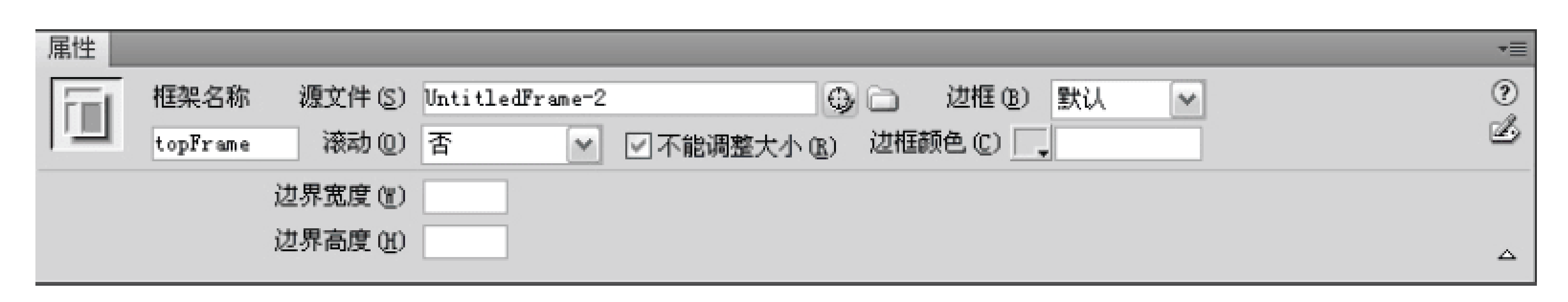


图 4-105　框架属性面板

框架属性面板中的参数说明如下。

- 框架名称：设置所选框架的名称。
- 源文件：设置框架中所显示网页的路径。
- 滚动：确定在没有足够的空间显示当前框架内容时，是否显示滚动条，有以下 4 个选项。
  - ☑ 是：在任何情况下都会显示滚动条。
  - ☑ 否：隐藏滚动条。
  - ☑ 自动：只有在当前框架中插入的内容超出显示范围后才显示滚动条。
  - ☑ 默认：浏览器的默认效果，相当于自动选项。
- 不能调整大小：设置在浏览网页时用户是否能够对框架边框进行调整。
- 边框：设置是否显示框架的边框。
- 边框颜色：设置框架边框的颜色。
- 边界宽度：设置框架中的网页与框架左、右边框的距离。
- 边界高度：设置框架中的网页与框架上、下边框的距离。

2. 查看或设置框架集属性

选中要设置属性的框架集，在属性面板中将显示该框架集的属性，如图 4-106 所示。



图 4-106　框架集属性面板

框架集属性面板中的参数说明如下。

- 边框：确定网页的框架边框是否可见。
- 边框宽度：设置框架集的边框宽度。
- 边框颜色：设置框架集的边框颜色。
- 值：设置选中框架的尺寸。
- 单位：设置选中框架的尺寸单位。

4.11.4　在框架中使用超链接

在框架式网页中制作超链接时，一定要设置链接的目标属性，为链接的目标文档指定显示窗口。在属性面板的“目标”下拉列表框中可以指定目标文件的显示窗口包括如下 4 个选项。

- _blank：放在新窗口中。
- _parent：放到父框架集或包含该链接的框架窗口中。
- _self：放在相同窗口中（默认窗口无须指定）。
- _top：放到整个浏览器窗口并删除所有框架。

另外，在“目标”下拉列表框中还会出现保存过的框架页名称。

4.12　站点的整理维护与上传

4.12.1　本地测试站点

一个网站从建立到投入使用通常要经过的步骤为：规划站点→构建本地站点和远程站点→站点的测试→站点的上传与发布。

网站制作完成后，还必须进行一项比较重要的工作，就是在本地对自己的网站进行测试，以免上传后出现错误不方便修改。本地测试包括网页的链接测试、网页在不同浏览器的兼容性测试以及下载时间和网页文件大小的测试。



1．检查站点链接和修复损坏链接

一个网站可由几十个、几百个甚至几千个网页组成，网页和网页之间依靠超链接来相互联系。如果一个网页中没有添加超链接或者是超链接添加错误，那么就无法找到该网页。所以在网站制作完成之后，首先要检查网站中的超链接是否正确。

利用 Dreamweaver 可以快速检查站点中网页的链接，避免出现链接错误。在“文件”面板中选中站点，单击鼠标右键，在弹出的快捷菜单中选择“检查链接”→“整个本地站点”命令，打开“链接检查器”选项卡，并以列表形式显示断掉的链接文件或者图片，如图 4-107 所示。

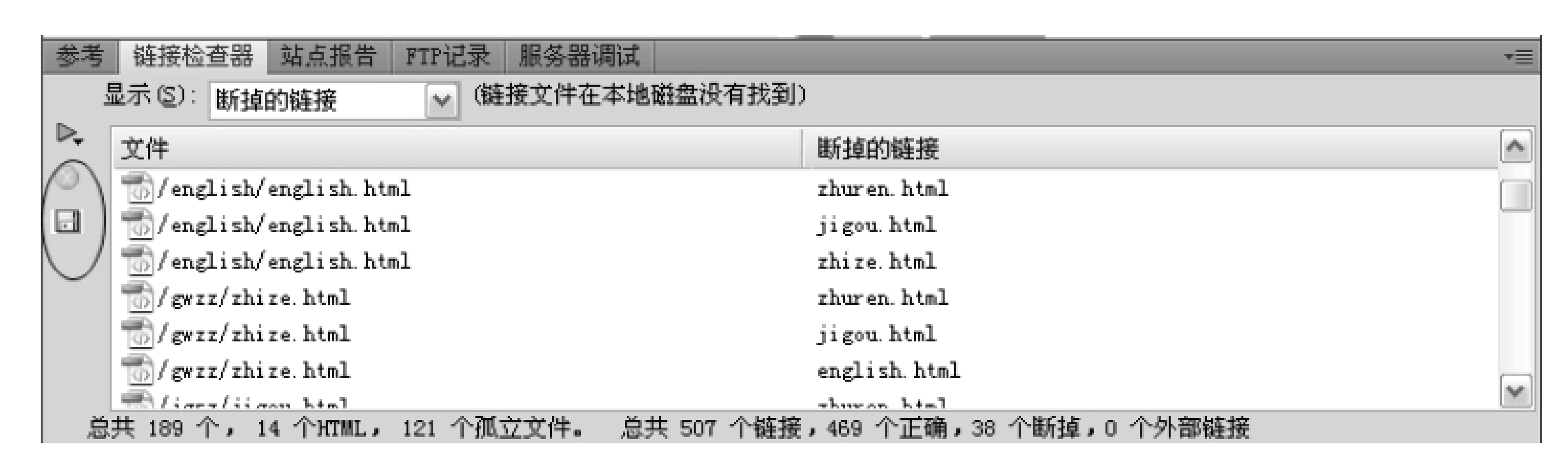


图 4-107 “链接检查器”选项卡

在“显示”下拉列表框中，可选择以下要检查的链接的类型。

- 断掉的链接：检查文档中是否存在断开的链接，这是默认选项。
- 外部链接：检查文档中的外部链接是否有效。
- 孤立文件：检查站点中是否存在孤立文件。所谓孤立文件，就是没有任何链接引用的文件。只有在检查整个站点链接的操作中才有效。

单击“保存报告”按钮（如图 4-107 椭圆框内所示），可将显示出来的错误链接保存。

要对断掉的链接进行修复，可以直接从链接检查器中修复损坏的链接文件或者图片，也可以从列表中将文件打开，然后在属性面板中修复。

在链接检查器中修复链接，可先选择无效链接栏下的链接文件名，然后输入正确的文件名和路径，或者单击文件夹图标，浏览并选定文件，修复完毕后按 Enter 键即可。

当然，也可以在链接检查器中双击打开存在损坏链接的文件，在文件中选中出错的图片或者链接，在属性面板中进行修改，最后保存文件即可。

链接修复后，该文件就会从链接检查器列表中消失，如果仍然存在，则表明还存在错误的链接。如果多个文件都有相同的中断链接，当对其中一个链接文件进行修改后，系统会询问是否修复余下的引用该文件的链接。

2．检查浏览器兼容性

不同浏览器的测试，就是在不同的浏览器和不同的版本下测试页面的运行和显示情况。Dreamweaver 能将测试出来的错误和可能出现的错误列出，然后可以根据系统给出的提示信息进行修改和处理，解决可能出现的问题，以免在浏览页面时出现错误。

检查浏览器兼容性功能可对文档中的 HTML 进行测试，检查是否有浏览器不支持的标签和属性，但对文档本身不进行任何更改。

选择“窗口”→“结果”→“浏览器兼容性”命令，打开如图 4-108 所示的“浏览器兼容性”选项卡，其中以列表形式显示网页中的兼容问题以及出现错误的具体行为和原因。

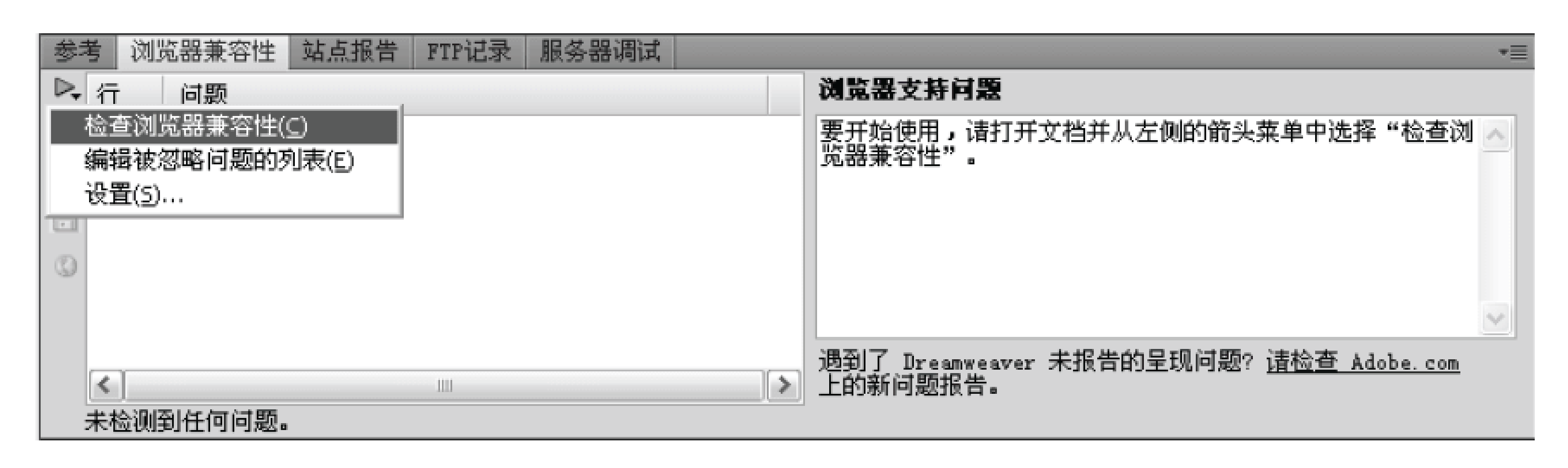


图 4-108 “浏览器兼容性”选项卡

4.12.2 申请主页空间和域名

1. 申请主页空间

一系列的站点测试工作完成之后，就可以开始申请空间和域名，为网站的正式发布作准备。

发布网站首要的任务就是在 Internet 的某些站点上申请免费的服务器空间（一般为几兆到上百兆不等），以便保存自己的网页。如果有条件，也可以向 ISP（Internet Server Provider，Internet 服务提供商，通常提供 Internet 接入等服务）申请需要交一定费用的服务器空间，这样的收费空间通常可以提供更多的服务（如数据库支持等）。

目前，多数知名的 Internet 站点都提供免费或收费的个人主页空间，如网易（http://www.163.com）等，按照这些站点上的提示可以很容易地进行申请网页空间的工作。

2. 申请注册域名

要想在网上建立服务器发布信息，则必须首先注册自己的域名，只有有了自己的域名才能让别人访问到自己的网站。同时，由于域名的唯一性，尽早注册是十分必要的。因此，为了在网上宣传自己的产品和服务，有头脑、有远见的企业和个人应当及时申请注册自己的域名。

在域名的选择时，需要注意以下 3 点。

- 避免难以记忆和过长的域名。
- 域名中尽量使用常用字符。
- 域名应该朗朗上口，便于记忆。

目前，申请域名有两种形式：一种是收费的，另一种是免费的。实际上，大多数域名是收费的，免费的域名已经越来越少了。免费域名只提供域名，不提供主页空间，因此这种域名实际上只提供一种转向功能，不能真正发布网页。

提供收费域名的 ISP 很多，如图 4-109 所示的是“中国互联”网站的收费域名申请页面。





图 4-109　“中国互联”网站的收费域名申请页面

4.12.3　发布站点

在完成了本地站点所有页面的设计之后，必须进行必要的测试工作，当网站能够稳定地工作后，就可以将站点上传到远程 Web 服务器上，使之成为真正的站点，这就是站点的发布。Dreamweaver 能够很轻松地完成站点的测试和发布。

网页设计好并在本地站点测试通过后，必须把它发布到 Internet 上形成真正的网站，否则网站形象仍然不能展现出去。要构建远程站点，必须知道 ISP 提供的主页空间是如何支持上传的。网页的上传一般是通过 FTP 软件工具连接 Internet 服务器进行上传的。FTP 软件很多，有 CutFtp、LeapFtp 等，也可以使用 Dreamweaver 的站点管理器上传网页。

Dreamweaver 内置了 FTP 上传功能，可以通过 FTP 实现在本地站点和远程站点之间的文件传输。

1．设置远程站点

当在本地机的硬盘上创建了本地站点之后，如果需要将本地站点传输到远程服务器上，就必须在传输站点之前，设置站点的服务器访问类型。设置远程站点信息的具体步骤如下：

（1）选择“站点”→“管理站点”命令，打开“管理站点”对话框。

（2）在“管理站点”对话框的站点列表中选择某个站点，单击“编辑”按钮进入站点定义对话框，选择“高级”选项卡，在“分类”列表框中选择“远程信息”选项，设置远程站点的服务器访问方式。

（3）从“访问”下拉列表框中选择服务器访问方式。在选择 FTP 连接服务器方式之后，Web 服务器信息栏中的选项及设置如图 4-110 所示。

（4）设置好各个参数后，用户可以单击“测试”按钮测试 FTP 远程站点是否连通。单

击“测试”按钮后，经过一段时间等待，出现测试成功的对话框，如图 4-111 所示。

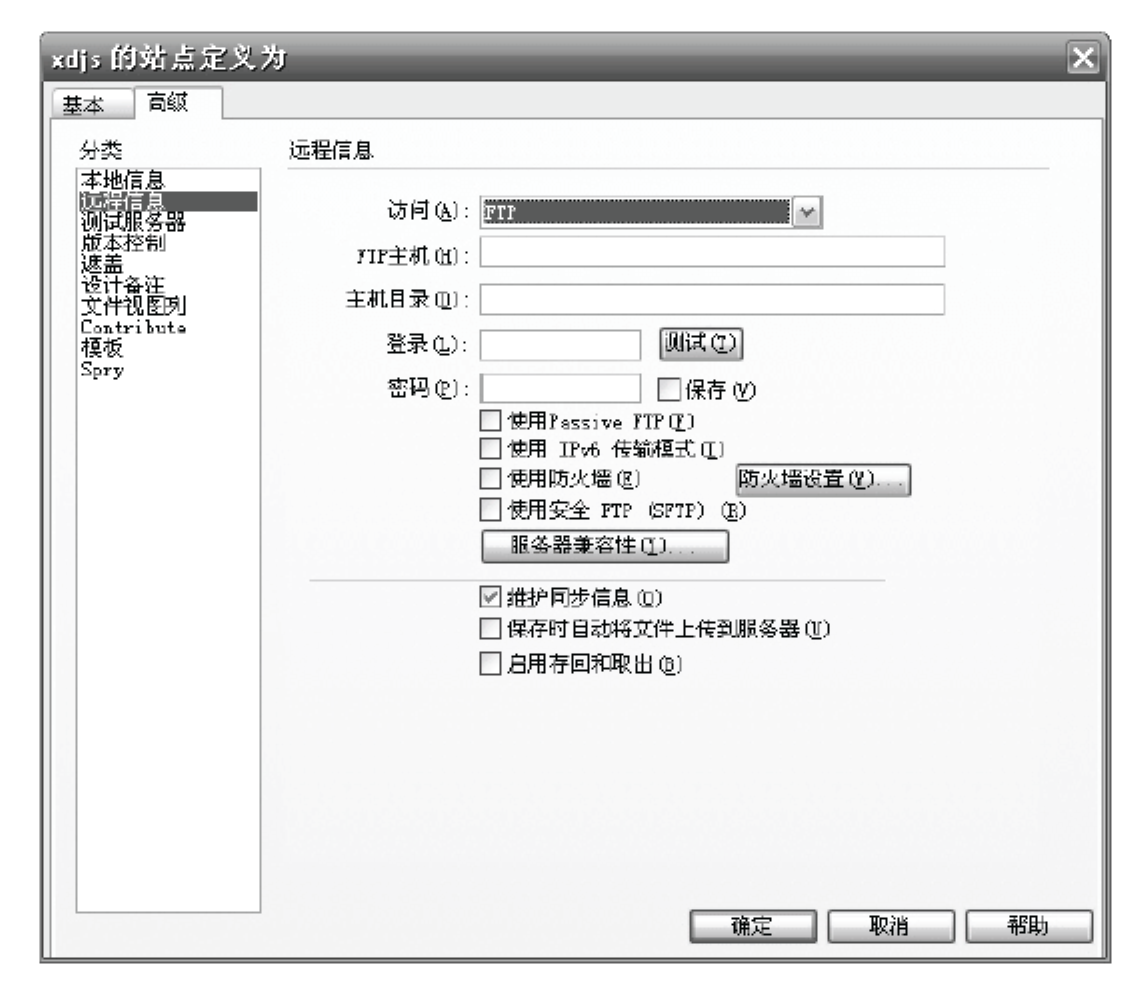


图 4-110　FTP 远程访问设置



图 4-111　测试 FTP 远程站点

（5）单击“确定”按钮进行确认并关闭该对话框，返回站点定义对话框，单击“确定”按钮，返回“管理站点”对话框，单击“完成”按钮，返回站点窗口。

2. 连接服务器

定义了远程站点后，还必须建立本地站点和 Internet 服务器的真正连接，才能真正构建远程站点。具体操作步骤如下：

（1）在站点窗口中显示要上传的本地站点。

（2）单击站点窗口上方的“连接”按钮，如图 4-112 所示。



图 4-112　“连接”按钮

（3）连接成功后，会在站点窗口的远程站点窗格中显示主机目录，它将作为远程站点根目录，同时原先的“连接”按钮转变为“断开连接”按钮。

3. 文件的上传和下载

在设置本地站点信息和远程站点信息后，就可以进行本地站点与远程站点间文件的上传及下载操作，具体操作步骤如下：

（1）在 Dreamweaver 中设置本地和远程服务器信息。

（2）将本地计算机连入 Internet。

（3）在站点管理窗口中打开要进行上传或下载文件操作的站点，将 Dreamweaver 与远程服务器接通，接通后，站点管理窗口左边的“远端站点”窗格中显示远程服务器中的文件目录。

（4）与远程服务器连通后，就可以在站点管理窗口中上传及下载站点文件。



4.12.4　网站的推广和宣传

目前，Internet 上的各种网站像天上的繁星、数不胜数。因此，对外发布的网站如果不进行宣传和推广，将无人知晓、无人问津。

网站的宣传和推广一般有两种途径：一种是通过传统媒体进行广告宣传，另一种是利用 Internet 自身的特点对外宣传。

可用来宣传的传统媒体包括电视、广播、报纸、广告牌、海报和黄页等，公司还可以在通信资料、产品手册和宣传品上印刷网站宣传信息。

在 Internet 上宣传网站的方法也是多种多样的，如可以将网址和网站信息发布到搜索引擎、网上黄页、新闻组和邮件列表上进行宣传推广，也可以与其他同类网站交换宣传广告。

4.13　小　　结

本章主要介绍了网页设计工具 Dreamweaver CS4 的操作方法，具体包括：站点的创建与管理、编辑文本内容、插入图像、网页链接、表格操作、创建 CSS 样式表/表单、插入声音、视频、Flash 动画元素等内容。通过对本章内容的学习，读者就可以自己设计一个具有多媒体效果的网站。

4.14　思　考　题

利用 Dreamweaver CS4 工具设计一个音乐网站。

第 5 章　Web 客户端脚本语言设计

网页是采用标记语言（HTML、XML）、客户端脚本语言（VBScript、JScript、JavaScript）和服务器端开发技术（ASP、JSP、PHP）共同设计而成的一种多媒体信息集合。第 2 章和第 4 章已经介绍了关于 HTML、XML 标记语言的基本概念和 Dreamweaver 网页制作工具的使用技巧，本章将重点介绍客户端脚本语言 JavaScript 的特点和编程方法。

5.1　客户端脚本语言简介

客户端指的是正在浏览、享受服务的一端。客户端脚本程序是指在浏览器中运行的程序，它由客户端脚本语言编写而成。该程序不需要事先编译，只需要通过浏览器自带的脚本引擎就能够对 HTML 文档中的脚本程序进行分析、识别、解释并执行。

5.1.1　常见的客户端脚本语言

脚本（Script）是使用一种特定的描述性语言，依据一定的格式编写的批处理文件。客户端脚本简单地说就是一条条的文字命令，这些文字命令是可以看到的（如可以用记事本打开查看、编辑），脚本程序在执行时，是由系统的一个解释器，将其一条条地翻译成机器可识别的指令，并按程序顺序执行。客户端脚本通常可以由应用程序临时调用并执行。各类客户端脚本被广泛地应用于网页设计中，因为它不仅可以减小网页的规模、提高网页浏览速度，而且可以丰富网页的表现，如动画、声音等。例如，当单击网页上的 E-mail 地址时能自动调用 Outlook Express 或 Foxmail 这类邮件软件，就是通过脚本功能来实现的。在浏览器的“Internet 选项”对话框中的“安全”选项卡中，单击“自定义级别”按钮，在安全设置列表的“活动脚本”栏中选中“禁用”、“启用”或“提升”单选按钮就可以轻松实现对客户端脚本的禁用和启用。常见的客户端脚本语言有 VBScript、JScript、JavaScript。

VBScript 是微软开发的一种脚本语言，可以看作是 VB 语言的简化版，与 VBA 的关系也非常密切，它具有原语言容易学习的特性，目前这种语言广泛应用于网页和 ASP 程序制作。由于 VBScript 可以通过 Windows 脚本宿主调用 COM，因而可以使用 Windows 操作系统中可以被使用的程序库，如它可以使用 Microsoft Office 的库，尤其是使用 Microsoft Access 和 Microsoft SQL Server 的程序库，当然它也可以使用其他程序和操作系统本身的库。

JScript 是 Microsoft 公司对 ECMA 262 语言规范（ECMAScript 编辑器 3）的一种实现。除了少数例外（为了保持向后兼容），JScript 完全实现了 ECMA 标准。JScript 是一种解释



型的、基于对象的脚本语言，有一定的局限性，例如，不能使用该语言来编写独立运行的应用程序，并且没有对读写文件的内置支持。此外，JScript 脚本只能在某个解释器或宿主上运行，如 Active Server Pages（ASP）、IE 浏览器或者 Windows 脚本宿主。

JavaScript 是根据 ECMAScript 标准制定的网页脚本语言，这个标准由 ECMA 组织发展和维护。ECMA-262 是正式的 JavaScript 标准，这个标准基于 JavaScript（Netscape）和 JScript（Microsoft）。Netscape（Navigator 2.0）的 Brendan Eich 发明了这门语言，从 1996 年开始，已经出现在所有的 Netscape 和 Microsoft 浏览器中。ECMA-262 的开发始于 1996 年，在 1997 年 7 月，ECMA 会员大会采纳了它的首个版本。JavaScript 最初的确是受 Java 启发而开始设计的，而且设计的目的之一就是“看上去像 Java”，因此语法上有很多类似之处，许多名称和命名规范也借自 Java。但是实际上，JavaScript 的主要设计原则源自 Self 和 Scheme，它与 Java 本质上是不同的，它与 Java 名称上的近似，是当时网景为了营销考虑与 Sun 公司达成协议的结果。其实从本质上讲 JavaScript 更像是一门函数式编程语言。而非面向对象的语言，它使用一些智能的语法和语义来仿真高度复杂的行为，其对象模型极为灵活、开放和强大，具有全部的反射性。

5.1.2　JavaScript 脚本语言的特点

JavaScript 是一种基于对象（Object）和事件驱动（Event Driven）并具有安全性能的脚本语言。JavaScript 认为文档和显示文档的浏览器都是由不同的对象组成的集合，这些对象具有一定的属性，可以对这些属性进行修改或计算。使用 JavaScript 的目的是与 HTML 超文本标记语言、Java 小程序一起实现在一个 Web 页面中链接多个对象，与 Web 客户交互作用，从而可以开发客户端的应用程序等。JavaScript 是通过嵌入或调入在标准的 HTML 语言中实现的，它可以弥补 HTML 语言的缺陷。JavaScript 脚本语言具有以下特点。

1. 是一种脚本编写语言

JavaScript 是一种脚本语言，它采用小程序段的方式实现编程。像其他脚本语言一样，JavaScript 同样已是一种解释性语言，它提供了一个简单的开发过程。它的基本结构形式与 C、C++、VB、Delphi 十分类似。但它不像这些语言一样，需要先编译，而是在程序运行过程中被逐行地解释。它与 HTML 标识结合在一起，从而方便用户的使用操作。

2. 基于对象的语言

JavaScript 是一种基于对象的语言，同时可以看作是一种面向对象的语言，这意味着它能运用自己已经创建的对象。因此，许多功能可以来自于脚本环境中对象的方法与脚本的相互作用。

3. 简单性

JavaScript 的简单性主要体现在：首先它是一种基于 Java 基本语句和控制流之上的简单而紧凑的设计，对于学习 Java 是一种非常好的过渡；其次它的变量类型是采用弱类型，并

未使用严格的数据类型。

4. 跨平台性

JavaScript 依赖于浏览器本身，与操作环境无关，只要能运行浏览器的计算机，并支持 JavaScript 的浏览器就可正确执行。

5. 安全性

JavaScript 是一种安全性语言，它不允许访问本地的硬盘，并不能将数据存入到服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，从而有效地防止数据的丢失。

6. 动态性

JavaScript 是动态的，它可以直接对用户或客户输入做出响应，无须经过 Web 服务程序。它对用户的响应是采用以事件驱动的方式进行的。所谓事件驱动，就是指在主页（Home Page）中执行了某种操作所产生的动作，称为事件（Event），如按下鼠标、移动窗口、选择菜单等都可以视为事件。当事件发生后，可能会引起相应的事件响应。

5.2 JavaScript 语言基础

JavaScript 是一种易学易用的脚本语言，是与用户动态交互的脚本开发，它扩展了 HTML 页面的功能，而不是开发大型复杂的程序，所以相对而言，JavaScript 的语法规则较少而且较为简单。作为一门编程语言，它有一套语法规则、关键字、语句和对象。为了运用 JavaScript 控制 HTML 页面上的对象，JavaScript 的代码必须与 HTML 代码结合在一起。将 JavaScript 嵌入 HTML 页面时，必须使用<script>标签，该标签使用格式如下：

```
<script language="JavaScript">
    //JavaScript 代码
</ script >
```

标签<script>通知浏览器，有脚本嵌入到标签中。

5.2.1 标识符与关键字

标识符是用来标识常量、变量、函数的名称符号，其命名必须以字母开始，由字母、数字、下划线组合而成。标识符命名要遵循如下的规则：

- 变量名不能与关键字冲突。
- 变量名必须以字母或者下划线（_）开头，不能用数字或者其他非字母字符作为变量名开头。



- 变量名中不能包含空格。
- JavaScript 是区分大小写的，所以给变量命名时要考虑大小写的问题。

关键字是 JavaScript 语言的保留字，用小写字符串表示语法含义的特殊字。如果书写有误，则会显示“JavaScript 脚本错误”的信息。常用的关键字有 var、if、switch、case、for、while、do、this 等。

5.2.2　数据类型

JavaScript 语言提供的数据类型包括字符串、数值（整数和实数）和布尔型。

- 字符串型：用单引号或双引号表示，“ABC”、“hello”、“您好”。

转义字符是用反斜杠开头的特殊字符型，常用在字符串中，表示不可显示的特殊字符。表 5-1 列出了常用的转义字符。

表 5-1　常用的转义字符

转义字符	描述	转义字符	描述
\b	表示退格	\"	表示双引号本身
\n	表示换行	\'	表示单引号本身
\f	表示换页	\\	表示反斜线
\t	表示 Tab 符号	\000	八进制数
\r	表示回车符	\xHH	十六进制数

- 数值型：整型，如 123；实型，如 123.23。
- 布尔型：逻辑真值用 true 表示，逻辑假值用 false 表示。

5.2.3　常量

常量是指在程序运行过程中，其值不变化的量。在脚本程序中常量可以直接给出，也可以用符号常量名间接表示。字符型常量用单引号或双引号表示，如“hello”、‘ok’；数值常量可以是整数（234）或实数（98.34）；若有些常量在整个程序中多次出现，可以用符号常量名定义（常量名一般用小写字母表示），如“const pi=3.141596”，用符号常量可以方便调试程序。

5.2.4　变量

所谓变量是指在程序运行过程中，其值在不断发生变化的量。每一个变量都有一个变量名（用标识符）表示。JavaScript 中采用弱类型变量，变量可以不做声明和类型说明，而在使用或赋值时确定类型即可。但为了形成良好的编程风格，变量应该采取先定义再使用的方法。JavaScript 中变量的定义用关键字 var 来说明。例如，定义一个名为 id 的变量：

```
< script  language ="JavaScript">
   var id;
</ script >
```

若定义多个变量名则用分号隔开。

5.2.5 运算符/表达式

JavaScript 语言提供了丰富的运算功能，包括算术运算、关系运算、逻辑运算和连接运算等，下面介绍各类运算符的含义。

1. 算术运算符

+（加）、-（减）、*（乘）、/（除）、%（取模）、++（递加）、--（递减）、|（按位或）、&（按位与）、<<（左移）、>>（右移）、~（取补）、-（取反）。

2. 比较运算符

<（小于）、>（大于）、<=（小于等于）、>=（大于等于）、==（等于）、!=（不等于）。

3. 逻辑运算符

&&（逻辑与）、||（逻辑或）、!（逻辑非）、^（逻辑异或）。

4. 赋值运算符

=（赋值）、+=（累加赋值）、-=（累减赋值）、*=（连乘赋值）、/=（连除赋值）、%=（取模赋值）。

5. 条件选择符

条件表达式？A:B（若条件表达式为 true，其值为 A；反之为 B）。

6. 连接运算符

+（连接两个字符串）。

表达式就是把常量、变量、函数用上述运算符连接在一起的式子。根据运算结果的不同，表达式也可以分为算术、关系、逻辑表达式。关系/逻辑表达式的值为布尔型（true、false）。

5.2.6 内置对象

JavaScript 中的内置对象包括数学（Math）对象、字符串（String）对象、日期（Date）对象、时间（Time）对象和数组（Array）对象等。

1. Math 对象

Math 对象封装了进行所有基本数学计算的功能和常量的属性和函数，如表 5-2 所示。



Math 对象不需要用 new 操作符创建对象，而是可以直接使用，所以又被称作静态的对象。调用方式为“Math.属性/函数(参数表)”。

表 5-2　Math 对象的属性和函数

属　　性	描　　述	函　　数	描　　述
PI	π 的值 3.142	random()	产生 0～1 之间的随机数
E	常数 e 的值	round(value)	value 的四舍五入取整
LN10	10 的自然对数值	abs(value)	value 的绝对值
LN2	2 的自然对数值	exp(value)	e 的 value 对数
SQRT2	2 的平方根	sqrt(value)	value 的平方根
SQRT1.2	0.5 的平方根	log(value)	value 的自然对数
LOG10E	以 10 为底的常数 E 的对数值	sin(value)	value 的正弦值
LOG2E	以 2 为底的常数 E 的对数值	cos(value)	value 的余弦值

例 5.1　创建简易计算器。

```
<html>
<head>
<script language="JavaScript">
alert(Math.abs(-1));//返回绝对值
alert(Math.round(3.8));//返回四舍五入之后的整数
alert(Math.random());//返回 0 和 1 之间的一个随机数
</script>
</head>
<body>
</body>
</html>
```

2．String 对象

String 对象封装了对字符串进行处理的属性和函数，它只有一个 length 属性，第 1 个字符位置值是 0。String 对象不需要用 new 定义，可以直接用 var 定义，如“varstr1="hello";”。String 对象封装了下列几个函数。

（1）定位查找字符——charAt(position)

功能：查找 String 对象中第 position 位置的字符。

例 5.2　定位查找字符示例。

```
<head>
<script language="JavaScript">
var   str1="hello";
var   findstr2=str1.charAt(4);
alert(findstr2);
</script>
</head>
```

```
<body>
</body>
</html>
```

结果：o。

说明：从 str1 串中查找第 5 个字符。该函数不适合在中文串中查找。

（2）从左查找字符位置——indexOf(substr)

功能：从 String 对象中查找 substr 字符串的位置。

例 5.3　从左查找字符位置示例。

```
<html>
<head>
<script language="JavaScript">
var   str1="hello";
var   findstr2=str1.indexOf("o");
alert(findstr2);
</script>
</head>
<body>
</body>
```

结果：4。

说明：从 str1 串中开始查找字符 o 的位置，其结果为 4。如果找不到，则返回-1。

（3）取子字符串——Substring(position1,position2)

功能：查找 String 对象中 position1 到 position2 位置之间的字符串。

例 5.4　取子字符串示例。

```
<html>
<head>
<script language="JavaScript">
var   str1="hello";
var   findstr2=str1.substring(2,5);
alert(findstr2);
</script>
</head>
<body>
</body>
</html>
```

结果：llo。

说明：从 str1 串的第 3 个字符开始查找 3 个字符串，结果为 llo。

（4）小写转换成大写——toUpperCase()

功能：将 String 对象小写字符转换成大写字符。



例 5.5　小写转换成大写示例。

```
<head>
<script language="JavaScript">
var   str1="hello";
var   findstr2=str1.toUpperCase();
alert(findstr2);
</script>
</head>
<body>
</body>
</html>
```

结果：HELLO。

（5）大写转换成小写——toLowerCase()

功能：将 String 对象大写字符转换成小写字符。

例 5.6　大写转换成小写示例。

```
<head>
<script language="JavaScript">
var   str1="HELLO";
var   findstr2=str1.toLowerCase();
alert(findstr2);
</script>
</head>
<body>
</body>
</html>
```

结果：hello。

（6）产生超链接字符——link(URL)

功能：形成超链接到 URL 上的字符串。

例 5.7　产生超链接字符示例。

```
<html>
<head>
<script language="JavaScript">
var   str1="HELLO";
var   linkstr2=str1.link("default.html");
alert(linkstr2);
</script>
</head>
<body>
</body>
</html>
```

linkstr2 等价于“<A HREF="default.html"> HELLO </A>”。

（7）产生瞄点字符——anchor()

功能：在网页中形成瞄点字符。

例 5.8　产生瞄点字符示例。

```
<head>
<script language="JavaScript">
var   str1="HELLO";
var   linkstr2=str1.anchor("Discount");
alert(linkstr2);
</script>
</head>
<body>
</body>
</html>
```

linktext2 等价于“<A NAME="Discount "> HELLO </A>”。

3．Date 对象

Date 对象封装了有关时间、日期的一些属性和函数，其含义如表 5-3 所示。

表 5-3　Date 日期对象的属性和函数

属　　性	描　　述	方　　法	描　　述	方　　法	描　　述
Year	年份	getYear()	获取年份值	setYear()	设置年份值
Month	月份	getMonth()	获取月份值	setMonth()	设置月份值
Day	日期	getDate()	获取日期值	setData()	设置日期值
Hours	小时	getHours()	获取小时值	setHours()	设置小时值
Minutes	分钟	getMinutes()	获取分钟值	setMinutes()	设置分钟值
Seconds	秒	getSeconds()	获取秒值	setSeconds()	设置秒值

Date 对象的定义格式有 3 种，下面分别进行介绍。

（1）格式 1：var day=new Date();

功能：获取计算机当期系统日期作为 day 对象的时间值。

例 5.9　格式 1 应用示例。

```
<html>
<head>
<script language="JavaScript">
var day=new Date();
alert(day);
</script>
</head>
<body>
</body>
</html>
```



（2）格式 2：var day=new Date(年,月,日);

功能：创建 day 对象为指定年份、日期、时间值，小时、分钟、秒为 0，如：

```
var Birthday=new Date(2001,4,23);
```

（3）格式 3：var day=new Date(年,月,日,小时,分,秒);

功能：创建一个具体年份的日期、时间作为 day 对象的值，如：

```
var meetday=new Date(2010,3,23,10,30,12);
```

例 5.10　显示系统日期。

```
<html>
<head>
<tatle> Data 对象的应用</tatle>
</head>
<script language=JavaScript>
var date=new Date();
var year=date.getYear();
var month=date.getMonth();
month=month+1;
var day=date.getDate();
var hours=date.getHours();
var minutes=date.getMinutes();
var seconds=date.getSeconds();
document.write("<pre>");
document.write("\n 当期日期为"+year+"年"+month+"月"+day+"日");
document.write("\n 当期时间为"+hours+"小时"+minutes+"分"+seconds+"秒");
document.write("</pre>");
</script>
 <body>
</body>
</html>
```

4．Array 对象

数组是一系列元素的有序集合，它的强大功能是不可替代的。在 JavaScript 中，可以使用 Array 对象来完成对数组的操作。Array 对象中的成员对象从 0 开始，其属性 length 表示数组的长度。Array 对象封装了对数组的排序、反序、取子数组等方法，其具体含义如表 5-4 所示。

表 5-4　Array 对象提供的方法

方　法	功　能
toString	输出所有数组元素
reveres()	使数组中的元素顺序反
slice(s,e)	返回一个从 s 到 e 之间的子数组
sort()	按字母顺序对数组进行排序

例 5.11　声明数组，定义 4 个学生姓名。

```
var student=new Aray(10);
student[0]="zhangping";
student[1]="liwei";
student[2]="wanghao";
student[3]="sunxia";
```

例 5.12　下拉菜单解释框示例。

```
<html>
<head>
<script language="JavaScript">
var messages = new Array(6);
messages[0] = "";
messages[1] = "163.com";
messages[2] = "126.com";
messages[3] = "QQ.com";
messages[4] = "yahoo.com.cn";
messages[5] = "sina.com ";
function messageReveal()
{
var messageindex = document.messageForm.messagePick.selectedIndex
document.messageForm.messageField.value = messages[messageindex];
}
document.write("<pre>");
document.write("\n 输出整个数组:"+messages.toString());
document.write("\n 输出数组第 3,4,5 元素:" +messages.slice(2,4));
document.write("</pre>");
</script
</head>
<body>
<form name="messageForm">
                    <select name="messagePick" onChange="messageReveal()">
                           <option value="0" selected>下拉菜单</option>
                           <option>163 邮箱</option>
                           <option>126 邮箱</option>
                           <option> QQ 邮箱</option>
<option>雅虎邮箱</option>
<option>新浪邮箱</option>
                         </select>
                         <br>
                         <p>
                  <textarea name="messageField" rows=6 cols=38 wrap=virtual
class="pt9"></textarea>
                       </form>
```



```
</body>
</html>
```

5.2.7　内置函数

JavaScript 中的内置函数如表 5-5 所示。

表 5-5　JavaScript 中的内置函数

函　数　名	功 能 描 述
escape()	对字符串进行编码
unescape()	对字符串进行解码
eval()	将字符串转换为实际代表的语句或运算
parseInt()	将其他类型的数据转换成整数
parseFloat()	将其他类型的数据转换成浮点数
isNaN()	判断一个表达式是否是字符

例 5.13　内置函数示例。

```
<html>
<head>
<script language="JavaScript">
var nu=prompt("请输入数据", 123);
var  str1=isNaN(nu);
if (str1)
{
 window.alert("不是数字字符串");
}else{
window.alert("数字字符串");
str1=parseInt(nu)*100;
str2="str1*100";
window.alert(eval(str2));}
</script>
</head>
<body>
</body>
</html>
```

如果输入的是数字，则打开一个提示窗口，显示“数字字符串”，然后进行计算；否则显示“不是数字字符串”。

5.2.8　事件

事件就是 Web 页面在浏览器处于活动状态时发生的各种事情，如浏览器加载、卸载一

个页面，用户单击鼠标、移动鼠标，以及按下键盘的某个键。事件调用函数就是当发生了某个事件之后去调用函数进行处理的方式。表 5-6 列出了 JavaScript 中常用的事件。

表 5-6　页面元素与对应的事件表

事件调用函数	何 时 触 发	支持的页面元素
onclick	鼠标单击时	所有元素
ondbclick	鼠标双击时	所有元素
onchange	显示的值改变时	表单元素
onfocus	窗口或元素获得焦点时	<body>和表单元素
onblur	窗口或元素失去焦点时	<body>和表单元素
onload	文档、图像或对象加载完毕时	<body>、<frameset>、<img>、<iframe>、<object>
onsubmit	表单提交时	<form>
onunload	文档卸载时	<body>、<frameset>

例 5.14　显示按钮单击事件。

```
<html>
<head>
<script language="JavaScript">
function clickme()
{
	alert("按钮被点击");
}
</script>
</head>
<body>
<input type="button" onclick="clickme()" value="点击我">
</body>
</html>
```

5.3　JavaScript 自定义函数

JavaScript 除了提供内置的函数外，还允许用户自定义具有计算功能的应用函数。

5.3.1　函数的定义

JavaScript 中的函数可以简单理解为一组语句，它们用来完成一系列工作。JavaScript 不区分函数和过程，因为它只有函数。



1．函数的定义

function 函数名(形参 1，形参 2，…，形参 N)

```
{
函数体(语句集)
}
```

2．函数的参数

在函数定义时确定参数，调用时将实参传递给形参。

3．函数返回值

可以使用 return 语句返回常量、变量或表达式等。

例 5.15　自定义判断闰年函数 isrunyear()。

```
function isrunyear(s)
{ var temp=false;
if ((n%400==0)||((n%4==0)&&(n%100!=0)))

{ temp=true;
}
return temp;
}
```

5.3.2　函数的调用

函数必须通过调用才能发挥作用。函数的调用格式为：

函数名（实参）

例 5.16　简单计算器。

```
<html>
<head>
<title> 简单计算器 </title>
<style>
input {width:40px;height:30px}
</style>
</head>
<body>
<script language="JavaScript">
function doact(key)
{
document.form1.t1.value+=key;
}
```

```
function docal()
{
document.form1.t1.value=eval(document.form1.t1.value);
}
</script>
<center>
<form name="form1" method="post" act="" style="background:#efefef;width:210px;">
<br>
<input tyle="text" name="t1" value="" style="width:190px;height:25px">
<br><br>
<input type ="button" value="+" onclick="doact(this.value)">
<input type ="button" value="-" onclick="doact(this.value)">
<input type ="button" value="*" onclick="doact(this.value)">
<input type ="button" value="/" onclick="doact(this.value)">
<br><br>
<input type ="button" value="1" onclick="doact(this.value)">
<input type ="button" value="2" onclick="doact(this.value)">
<input type ="button" value="3" onclick="doact(this.value)">
<input type ="button" value="4" onclick="doact(this.value)">
<br><br>
<input type ="button" value="5" onclick="doact(this.value)">
<input type ="button" value="6" onclick="doact(this.value)">
<input type ="button" value="7" onclick="doact(this.value)">
<input type ="button" value="8" onclick="doact(this.value)">
<br><br>
<input type ="button" value="9" onclick="doact(this.value)">
<input type ="button" value="0" onclick="doact(this.value)">
<input type ="button" value="." onclick="doact(this.value)">
<input type ="button" value="=" onclick="docal()">
<br><br>
<input type ="button" value="pi" onclick="doact(Math.PI)">
<input type ="button" value="sqrt" onclick="t1.value=Math.sqrt(t1.value)">
<input type ="button" value="1/x" onclick="t1.value=1/(t1.value)">
<input type ="button" value="c" onclick="t1.value=''">
<br><br>
</form>
</center>
</body>
</html>
```

5.4 JavaScript 控制语句

JavaScript 提供了 if…else、switch…case、for、while、continue 等多种形式的语句用于



控制程序的执行顺序，这些语句的功能和 C++语句功能相似。下面简要介绍 JavaScript 控制语句的功能及应用。

5.4.1　条件语句 if…else

条件语句 if…else 的格式如下：

```
if (条件表达式)
     {
      语句块 1;
      }
 < else>   //可选
     {
      语句块 2;
}
```

功能：如果条件表达式为 true，则执行语句块 1；否则执行语句块 2。其逻辑结构如图 5-1 所示。

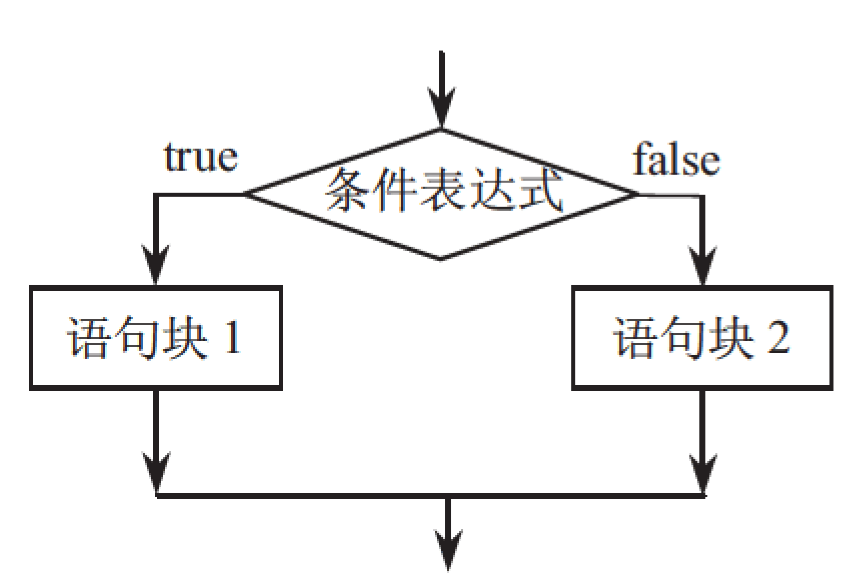


图 5-1　条件语句的逻辑结构

例 5.17　问候语。

```
<html>
<head>
<script language="JavaScript">
var nowtime=new Date();
var now_hour=nowtime.getHours();
if ((now_hour>=0 )&& (now_hour<6))
{ window.alert("熬夜了吗");
}
if ((now_hour>=6 )&& (now_hour<8))
{ window.alert("早上好");
}
if ((now_hour>=8 )&& (now_hour<11))
{ window.alert("上午好");
}
if ((now_hour>=11 )&& (now_hour<13))
```

```
{ window.alert("午饭吃了吗");
}
if ((now_hour>=13 )&& (now_hour<17))
{ window.alert("下午好");
}
if ((now_hour>=17 )&& (now_hour<24))
{ window.alert("晚上好");
}
</script>
</head>
<body>
</body>
</html>
```

5.4.2 选择语句 switch…case

选择语句 switch…case 的格式如下：

```
switch (条件表达式)
  {case 值 1:语句 1;
            break;
   case 值 2:语句 2;
            break;
            …
   case 值 n:语句 n;
            break;
   default:语句 n+1;
  }
```

功能：根据条件表达式的不同取值，分别执行其对应的语句；否则执行 default 后面的语句。其逻辑结构如图 5-2 所示。

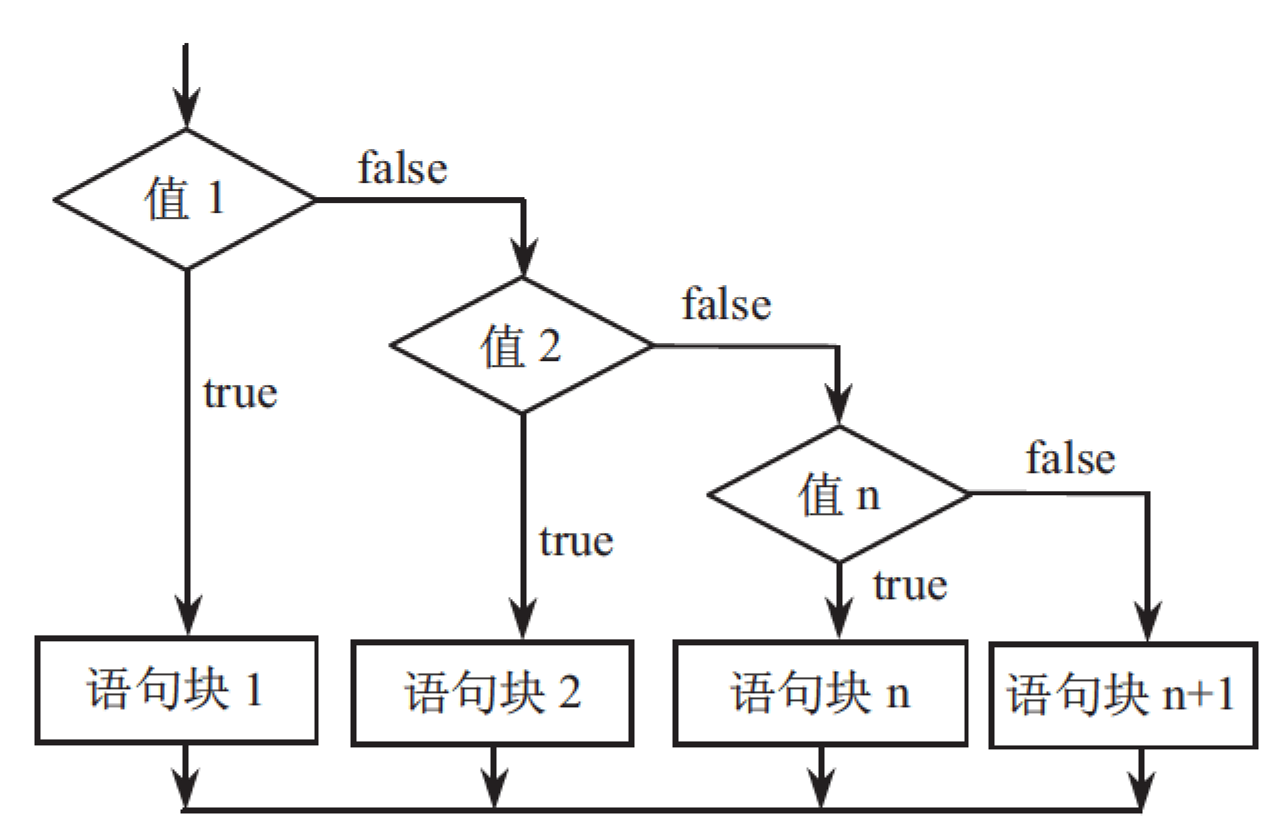


图 5-2 选择语句的逻辑结构



例 5.18　输入年、月数值，输出对应的天数。

```
<html>
<head>
<script language="JavaScript">
function isrunyear(n)
{ var temp=false;
if ((n%400==0)||((n%4==0)&&(n%100!=0)))
{ temp=true;
}
return temp;
}
function havedays(theyear,themonth)
{
var tempresult=0;
switch(themonth)
  {
  case 1:
  case 3:
  case 5:
  case 7:
  case 8:
  case 10:
  case 12:tempresult=31;break;
  case 4:
  case 6:
  case 9:
  case 11:tempresult=30;break;
  case 2:tempresult=(isrunyear(theyear)?29:28);break;
}
return tempresult;
}
var year=prompt("输入年份",2002);
var month=prompt("输入月份",12);
var temp1=havedays(parseInt(year),parseInt(month));
document.write(temp1);
</script>
</head>
<body>
</body>
</html>
```

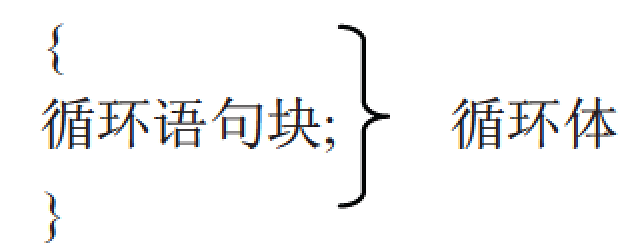
5.4.3　循环语句

循环语句的格式有 for 语句、while 语句和 do-while 语句 3 种，下面分别进行介绍。

（1）for 语句

格式如下：

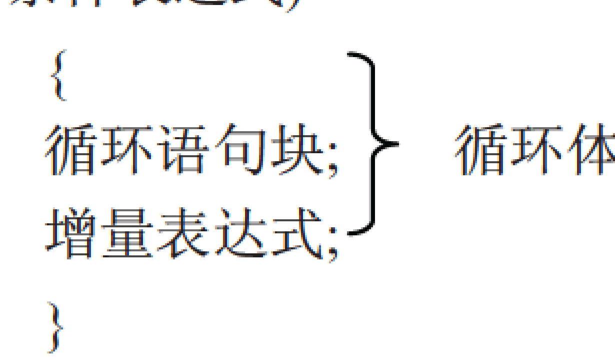
for (初始表达式;循环条件表达式;增量表达式)



（2）while 语句

格式如下：

while (循环条件表达式)



}

（3）do-while 语句

格式如下：

```
do{
        循环语句块;     } 循环体
        增量表达式;
}while (循环条件表达式)
```

功能：当循环条件表达式为 true 时，执行循环体；否则退出循环体，继续执行后续语句。for、while 语句的逻辑结构如图 5-3 所示。

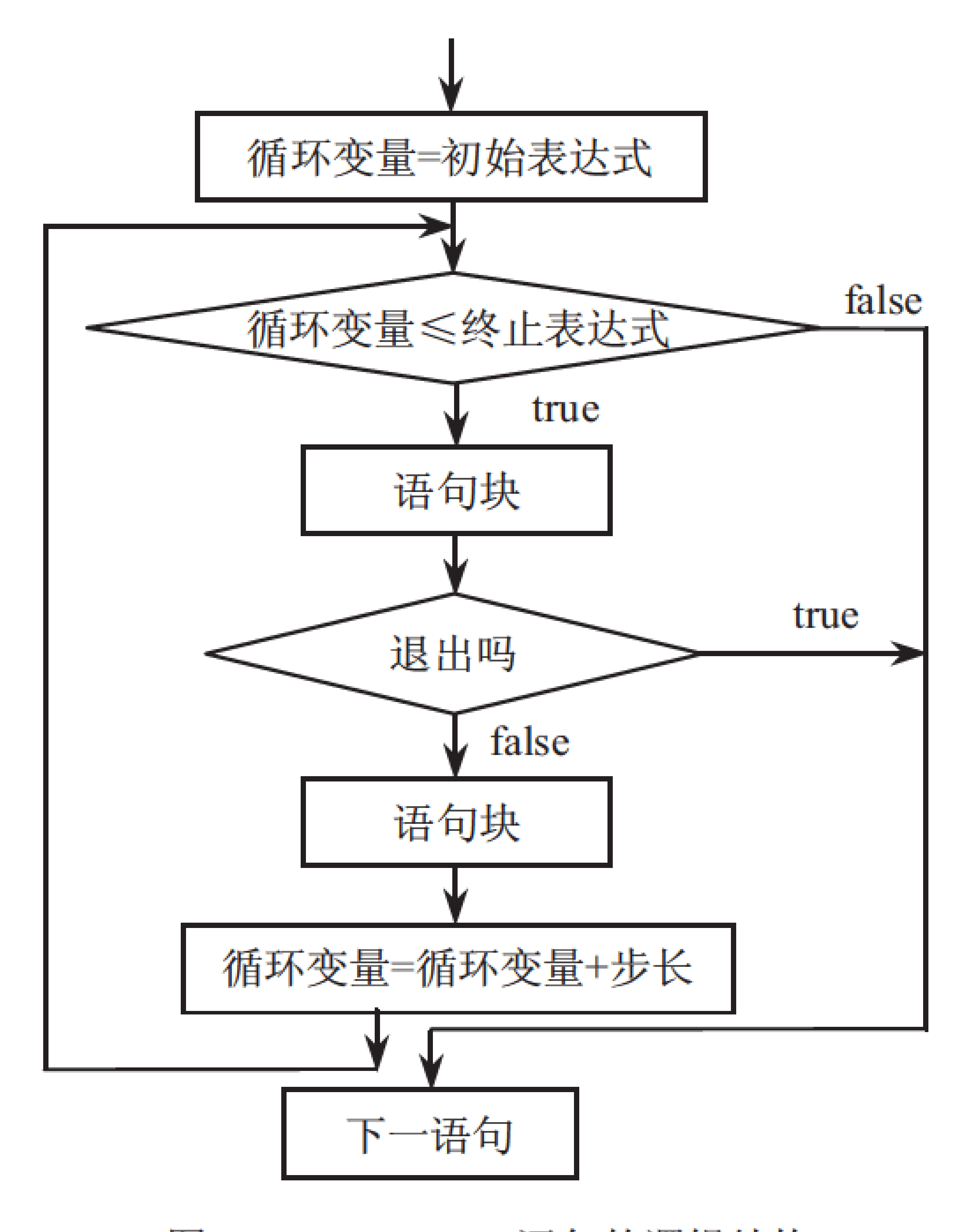


图 5-3　for、while 语句的逻辑结构



例 5.19　中文日历。

```
<html>
<title>  中文日历 </title>
<head>
<script language="JavaScript">
var str1="日,一,二,三,四,五,六";
var weekarry=str1.split(",");
var today=new Date();
var yy=today.getYear();
var mm=today.getMonth()+1;
var dd=today.getDate();
var ww=weekarry[today.getDay()];
function isrunyear(n)
{ var temp=false;
if ((n%400==0)||((n%4==0)&&(n%100!=0)))
{ temp=true;
}
return temp;
}
function havedays(theyear,themonth)
{
var tempresult=0;
switch(themonth)
  {
  case 1:
  case 3:
  case 5:
  case 7:
  case 8:
  case 10:
  case 12:tempresult=31;break;
  case 4:
  case 6:
  case 9:
  case 11:tempresult=30;break;
  case 2:tempresult=(isrunyear(theyear)?29:28);break;
}
return tempresult;
}
var runyear="";
if (isrunyear(yy))
{runyear="闰年";
}
document.write("<div style='width:200px;' align='center'>");
document.write((yy+"年"+mm+"月"+runyear+"<br>星期"+ww+"<br>").fontsize("6"));
document.write("<span style='font-size:72px'>"+dd+"</span>");
```

```
document.write("<table width='100%'>");
document.write("<tr style='background:#efefef'>");
for (var i=0;i<=6;i++)
 {document.write("<td>"+weekarry[i]+"</td>");
}
document.write("</tr>");
var firstday=new Date(yy+"/"+mm+"/1");
var fweek=firstday.getDay();
var daynum=havedays(yy,mm);
for (var j=1;j<=6;j++)
{
document.write("<tr>");
for (var i=0;i<=6;i++)
{
    theday=7*(j-1)+i+1-fweek;
if (((j==1)&&(i<fweek))||(theday>daynum))
{
document.write("<td> ></td>");
}
else
{
switch(i)
{
case 0:{document.write("<td>"+(""+theday).fontcolor("#ff0000")+"</td>");
break;}
case 6:{document.write("<td>"+(""+theday).fontcolor("#00cc00")+"</td>");
break;}
default :document.write("<td>"+theday+"</td>");
}
}
}
document.write("</tr>");
}
document.write("</table>");
document.write("</div>")
</script>
</head>
<body>
</body>
</html>
```

5.4.4 其他语句

（1）for-in 语句

格式如下：



for (变量 in 对象)
{ 代码块;}

功能：每次循环将对象的属性依次赋予循环变量。

（2）break 语句

格式如下：

break;

功能：跳出循环，继续执行该循环之后的代码。

（3）continue 语句

格式如下：

continue;

功能：提前结束当前循环，转向下一次循环。

例 5.20 显示对象属性。

```
<html>
<head>
</head>
<script language="JavaScript">
 var objects={user:"zhang",age:23,qq:"1234****",email:"ppp.@126.com"};
  for (var example in objects)
  {
if (example=="user")
 { window.alert("用户名="+objects[example]);continue ;}
if (example=="age")
 { window.alert("年龄="+objects[example]);}
  if (example=="qq")
 { window.alert("QQ 号="+objects[example]);}
 if (example=="email")
  {break;}
window.alert("下次循环");
}
</script>
<body>
</body>
</html>
```

5.5 浏览器对象模型 BOM

浏览器对象模型 BOM 是指当用户打开浏览器时，浏览器中的 JavaScript runtime engine 在内存中自动创建一组对象，用于对浏览器及 HTML 文档对象模型中数据的访问和操作。

5.5.1 BOM 层次结构

浏览器对象模型的结构如图 5-4 所示。

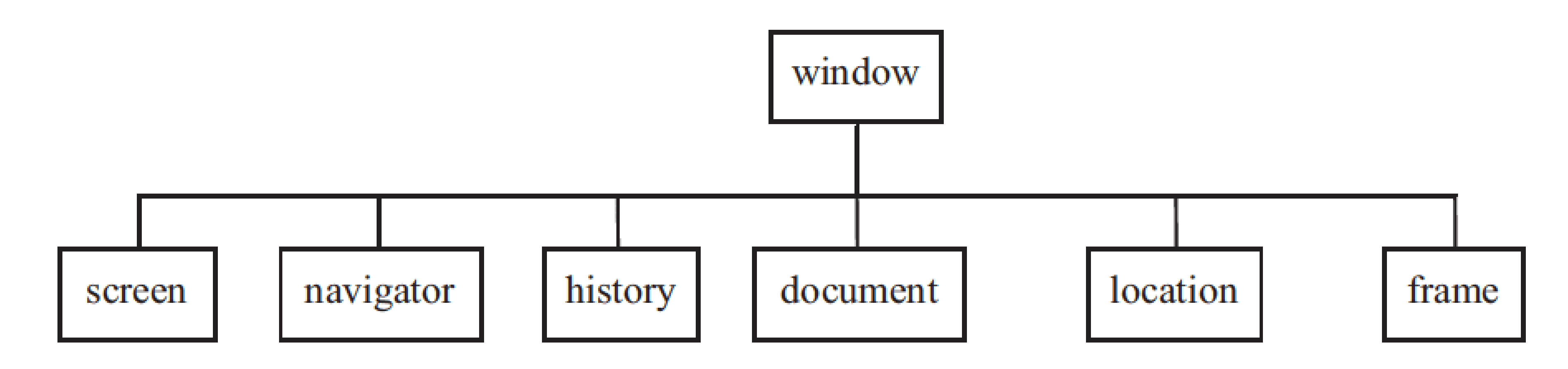


图 5-4 浏览器对象模型的结构

浏览器对象常见属性及方法如表 5-7 所示。

表 5-7 浏览器对象常见属性及方法

属 性	描 述	方 法	描 述
document	对话框中显示的当前文档	alert()	弹出一个警示对话框
frames	表示当前对话框中的 frames 对象集合	prompt()	弹出一个提示对话框
location	指定当前文档的 URL	confirm()	在确认的对话框中显示指定的字符串
name	对话框的名称	open()	在新的浏览器中打开对话框并显示有 URL 引用的文档，并设置创建对话框的属性
status	显示状态栏中的当前信息	close()	关闭被引用的对话框
navigator	表示浏览器对象，用于获取与浏览器有关的信息	print()	相当于浏览器工具栏中的“打印”按钮
self	表示当前对话框	navigate()	使用对话框显示 URL 指定的网页
closed	表示当前对话框是否关闭的逻辑值	status()	状态栏
opener	表示打开当前对话框的父对话框	resize()	设置对话框的大小
screen	表示用户屏幕，提供屏幕尺寸、颜色深度等信息	moveto()	将对话框移动到指定的位移量处
top	表示最顶层的浏览器对话框	focus()	被引用的对话框放在所有打开对话框的前面
parent	表示包含当期对话框的父对话框	blur()	被引用的对话框放在所有打开对话框的后面

浏览器对象的引用方式有以下两种：

- 逐层引用

例如，在引用 forms 对象时，使用 window.document.forms。

- 数组型浏览器对象的引用

在文档对象模型中，有些对象属于数组型对象，如 document 对象下一层的 images、links、forms 等对象，在引用这种数组对象时，可以使用对象在数组中的位置（下标）来引用。



例如，window.document.forms[0]表示引用文档中的第一个表单。

Window 对象作为文档对象模型中的最顶层对象，JavaScript 认为它是默认的，因此可以不写出来，如 window.document.forms 可以写成 document.forms。

下面重点介绍几个浏览器对象。

1．窗口对象（Window）

Window 对象处于对象层次的最顶端，它提供了处理浏览器窗口的方法和属性。该对象主要包括下列方法：

- open("url", ["window name"[,"window feature"]])

功能：打开具有指定名称的新窗口，并加载给定 URL 所指定的文档；如果没有提供 URL，则打开一个空白文档。

参数说明如下。

- ☑ url：要打开窗口的 URL 地址。
- ☑ window name：打开窗口的名称，可选项。
- ☑ window features：新窗口的外观，可选项。其参数有 height（窗口的高度）、width（窗口的宽度）、menubar（是否有菜单）、scrollbars（是否有滚动条）及 resizable（窗口大小是否可以改变）。

- close()方法

功能：关闭当前窗口。

例 5.21　打开指定窗口示例。

```
<html>
<head>
</head>
< script language="javascript">
function openwindow( ) {
  window.status="系统当前状态：您正在注册用户......";
  if (window.screen.width == 1024 && window.screen.height == 768)
    open("register.html", " 注 册 窗 口 ", "toolbars=0, location=0, statusbars=0, menubars=0, width=700,height=550,scrollbars=1");
  else
        window.alert("请设置分辨率为 1024×768，然后再打开");    }
function closewindow( )
{
    if(window.confirm("您确认要退出系统吗？"))
        window.close( );
}
</script >
<body>

<form>
```

```
<input type="button" name="regButton" value=" 用户注册 "
            onclick="openwindow( )">
<input type="button" name="exitButton" value=" 退 出 "
            onclick="closewindow( )">
</form>
</body>
</html>
```

- alert()方法

功能：弹出一个显示信息的警示对话框，如图 5-5 所示。

- confirm()方法

功能：弹出一个带有提示信息的确认对话框，如图 5-6 所示。

- prompt()方法

功能：弹出一个带有输入信息的对话框，如图 5-7 所示。



图 5-5　警示对话框



图 5-6　确认对话框



图 5-7　输入信息对话框

2. 位置对象（Location）

Location 对象提供了与当前打开的 URL 一起工作的方法和属性，它是一个静态的对象。Location 对象的属性及方法如表 5-8 和表 5-9 所示。

表 5-8　Location 对象属性

属 性 名	说 明
host	设置或检索位置或 URL 的主机名和端口号
hostname	设置或检索位置或 URL 的主机名部分
href	设置或检索完整的 URL 字符串

表 5-9　Location 对象方法

属 性 名	说 明
assign("url")	加载 URL 指定的新的 HTML 文档
reload()	重新加载当前页
replace("url")	通过加载 URL 指定的文档来替换当前文档

3. 历史对象（History）

History 对象提供了与历史清单有关的信息，它提供了下列 3 种常用的方法。

- back()方法：加载 History 列表中的上一个 URL。
- forward()方法：加载 History 列表中的下一个 URL。



- go("url" or number)：加载 History 列表中指定的 URL，或要求浏览器移动指定的页面数。

例 5.22　History 对象应用示例。

```
<FORM name="form1" method="post" action="">
…
<TD width="4%"><A href="javascript: history.back( )">返回 </A></TD>
<TD width="4%"><A href="javascript: history.forward( )">前进</A></TD>
<TD width="4%"><A href="javascript: location.reload( )">刷新</A></TD>
<TD width="6%"><A href="../index.html">首页</A></TD>
跳转到其他板块
<SELECT name="selTopic"  id="selPTopic" onChange="javascript: location=this.value">
     <OPTION value="news.html" selected="selected">新闻</OPTION>
      <OPTION value="gard.html">网上谈兵</OPTION>
      <OPTION value="IT.html">茶馆</OPTION>
     <OPTION value="education.html" selected >教育</OPTION>
</SELECT>
</FORM>
```

4．文档对象（Document）

Document 对象代表浏览器窗口中的文档，可以用来处理文档中包含的 html 元素，如表单、图像、超链接等。Document 对象包含了与文档元素（elements）一起工作的对象，它将这些元素封装起来供编程人员使用。有关文档对象的属性及方法的详细内容可参见本章第 5.6 节的相关内容。

5.5.2　浏览器对象的应用

例 5.23　利用浏览器对象实现在浏览器状态栏中从右向左自动滚动“计算机”文字。

```
<html>
<head>
</head>
<script language="JavaScript">
function scrollit_r2l(seed)
{ var m1  = "计算机" ;
 var m2  = "" ;
        var msg=m1+m2;
        var out = " ";
        var c = 1;
 var speed  = 100;
if (seed > 100)
```

```
{                         seed-=2;
                         var cmd="scrollit_r2l(" + seed + ")";
                    timerTwo=window.setTimeout(cmd,speed);}
      else if (seed <= 100 && seed > 0)
       {                          for (c=0 ; c < seed ; c++)
                                  {            out+=" ";}
           out+=msg;              seed-=2;
     var cmd="scrollit_r2l(" + seed + ")";
     window.status=out;
  timerTwo=window.setTimeout(cmd,speed); }
         else if (seed <= 0)
{                         if (-seed < msg.length)
                           {
                                  out+=msg.substring(-seed,msg.length);
                                  seed-=2;
                                  var cmd="scrollit_r2l(" + seed + ")";
                                  window.status=out;
           timerTwo=window.setTimeout(cmd,speed);}
           else {                        window.status=" ";
                          timerTwo=window.setTimeout("scrollit_r2l(100)",speed);
}
}
}
scrollit_r2l(100);
</script>
</script>
<body>
</body>
</html>
```

5.6 文档对象模型 DOM

当在浏览器中打开一个网页时，网页中的每个标记都在内存中创建了一个对象，由这些对象按照树型结构组成的模型就称为文档对象模型 DOM（Document Object Model）。

5.6.1 DOM 模型结构

按照 W3C DOM 规范，DOM 是一种与浏览器、平台、语言无关的接口，其结构如图 5-8 所示。



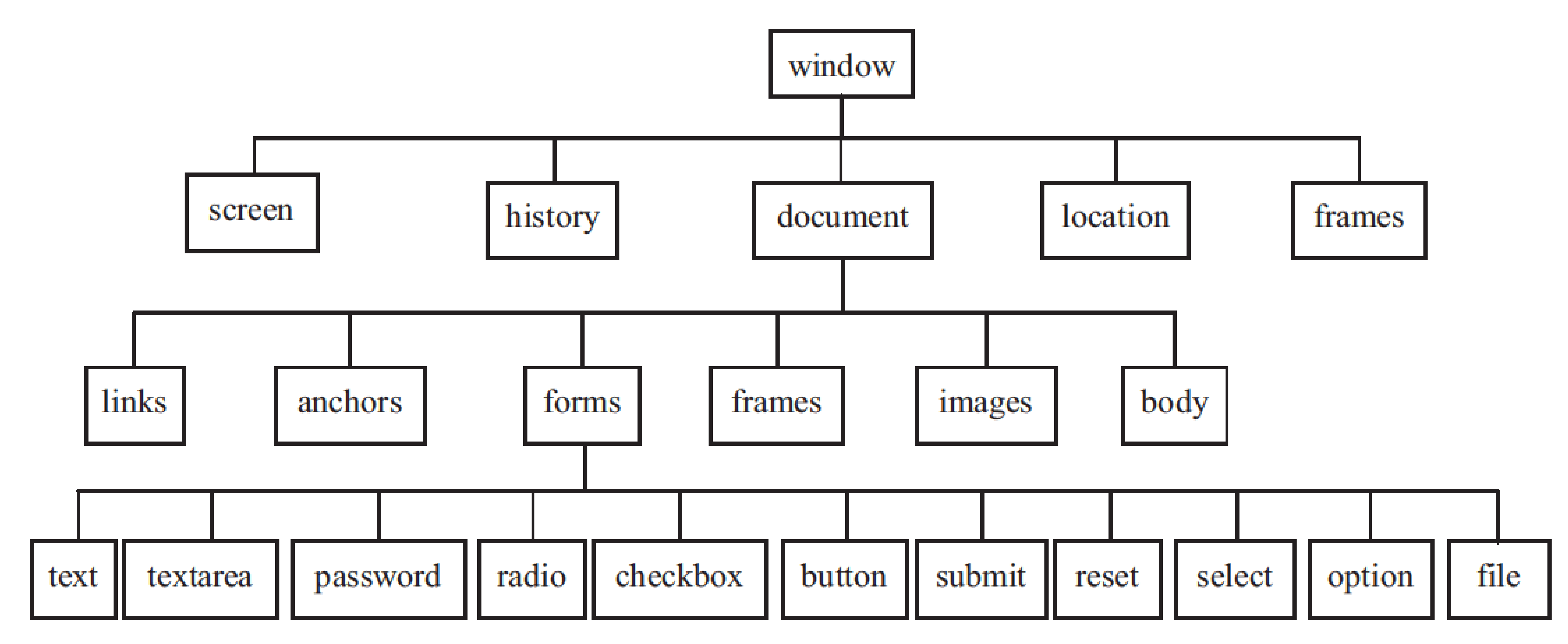


图 5-8　DOM 模型结构图

文档对象的常用属性和方法如表 5-10 所示。

表 5-10　文档对象的常用属性和方法

属　性	描　述	方　法	描　述
clinkcolor	链接文字的颜色	Close()	文档的输出流
bgcolor	文档的背景颜色	Open()	打开一个文档的输出流并接收 write 和 writeln 方法创建页面的内容
cookie	表示 cookie 的值	Write()	向文档中写入 HTML 或 JavaScript 语句
URL	获取或设置 URL	Writeln()	向文档中写入 HTML 或 JavaScript 语句，并以换行符结束
body	当前文档主体对象	creatElement	创建一个 HTML 标记
fileSize	当前文件的大小	getElementByld()	获取指定 ID 的 HTML 标记

5.6.2　文档对象的应用

例 5.24　显示当前系统时间。

```
<html>
<head>
</head>
<script language="JavaScript">
tmpDate = new Date();
date = tmpDate.getDate();
month= tmpDate.getMonth() + 1 ;
year= tmpDate.getYear();
document.write(year);
document.write("年");
document.write(month);
```

```
document.write("月");
document.write(date);
document.write("日");
</script>
<body>
</body>
</html>
```

例 5.25　实现全选效果。

```
<html>
<head>
</head>
<SCRIPT language="javascript">
function checkAll(boolValue ) {
    var allCheckBoxs=document.getElementsByName("isBuy") ;
    for (var i=0;i<allCheckBoxs.length ;i++)
        {
         if(allCheckBoxs[i].type=="checkbox")
           allCheckBoxs[i].checked=boolValue ;
        }                                              }
</SCRIPT>
<TR>
     <TD width="6%"><A href="javascript: checkAll(true)">全选  </TD>
     <TD width="6%"><A href="javascript: checkAll(false)">全不选
                </A></TD>
     <TD width="88%"><IMG src="images/top.jpg" width="845"
                height="18"></TD>
  </TR>
<body>
</body>
</html>
```

例 5.26　验证表单。

```
<html>
<head>
</head>
<SCRIPT LANGUAGE = "JavaScript">
  function checkUserName(){       //验证用户名
            var fname = document.myform.txtUser.value;
        if(fname.length != 0){
            for(i=0;i<fname.length;i++){
                var ftext = fname.substring(i,i+1);
                 if(ftext < 9 || ftext > 0){
                 alert("名字中包含数字  \n"+"请删除名字中的数字和特殊字符");
                 return false                 }        }      }
```



```
            else{       alert("请输入:名字);
                                document.myform.txtUser.focus();
                                return false        }
     return true;        }
    function passCheck(){ //验证密码
    var userpass = document.myform.txtPassword.value;
    if(userpass == ""){
    alert("未输入密码 \n" + "请输入密码");
    document.myform.txtPassword.focus();
    return false;    }
    if(userpass.length < 6){
    alert("密码必须多于或等于 6 个字符。\n");
    return false;    }
    return true;    }
    </SCRIPT>
    <FORM name="myform" onSubmit="return validateform( )" method="post"action="reg_success.htm" >……
    <INPUT name="registerButton" type="submit" id="registerButton" value="  登  录  "  >
    <FORM name="myform" method="post" action="reg_success.htm" onSubmit="return checkEmail( )">  ……
    <INPUT name="registerButton" type="submit" id="registerButton" value="  注  册  "  >
    <body>
    </body>
    </html>
```

5.7　Ajax 技术

Ajax（Asynchronous JavaScript and XML）是多种技术的综合，它使用 XHTML 和 CSS 标准化呈现，使用 DOM 实现动态显示和交互，使用 XML 和 XSTL 进行数据交换与处理，使用 XMLHttpRequest 对象进行异步数据读取，使用 JavaScript 绑定和处理所有数据。Ajax 并不是一项新技术，而是几个现有技术的新组合，而且它的发展也得益于几家大的互联网企业的率先应用，可以说 Ajax 已成为 Web 开发的重要武器。

5.7.1　Ajax 异步模型

传统的 Web 应用允许用户填写表单（Form），当提交表单时就向 Web 服务器发送一个请求。服务器接收并处理传来的表单，然后返回一个新的网页。这个做法浪费了许多带宽，因为在前后两个页面中的大部分 HTML 代码往往是相同的。由于每次应用的交互都需要向服务器发送请求，应用的响应时间就依赖于服务器的响应时间，这导致了用户界面的响应比本地应用慢得多。

与此不同是 Ajax 应用可以仅向服务器发送并取回必需的数据，它使用 SOAP 或其他一

些基于 XML 的 Web Service 接口，并在客户端采用 JavaScript 处理来自服务器的响应。因为在服务器和浏览器之间交换的数据大量减少，结果就能看到响应更快的应用。同时很多的处理工作可以在发出请求的客户端机器上完成，所以 Web 服务器的处理时间也减少了。

Ajax 使用 Ajax/JavaScript 提交模型。在此模型中，使用 JavaScript 截取事件调用，当该事件发生时（如用户单击“提交”按钮），提交的数据传递给对应的脚本，然后有脚本发起对服务器的调用，因为脚本可以立即响应事件且不必等待数据的提交，所以从服务器返回的数据也不必马上显示给用户，而脚本也不必像以前一样被动等待服务器的响应。JavaScript 使用 XMLHttpRequest 对象的 POST 或 GET 方法，将数据封装到 URL 或 Request 中，向服务器提交数据。在 PHP 中，可以使用$_GET、$_POST、$_REQUEST 收集客户机传上来的数据；在 ASP、ASP.NET 中，可以使用 QueryString、Form 或 Params 收集客户机传上来的数据。服务器获得数据后，就可以按照自己的方式进行处理，然后将处理结果返回给客户端。

1. 创建 XMLHttpRequest 对象

XMLHttpRequest 对象最初是作为 IE 5 中的一个 ActiveX 控件出现的，随后 Mozilla 1.0、Netscape 7、Safari 1.2 都将它纳入其中。XMLHttpRequest 对象在 IE 浏览器和非 IE 浏览器中的实现方法不同。下面分别介绍两种不同的创建代码。

在 Mozilla、Netscape、Safari 浏览器中，创建 XMLHttpRequest 对象的 JavaScript 代码为：

```
xmlhttp_request=new XMLHttpRequest();
```

在 IE 浏览器中，创建 XMLHttpRequest 对象的 JavaScript 代码为：

```
xmlhttp_request = new ActiveXObject("Microsoft.XMLHTTP");//IE2.0
xmlhttp_request = new ActiveXObject( "Msxml2.XMLHTTP");//IE2.0 以上
```

也可以用创建一个函数 createXmlReq()将两者一并考虑：

```
var httpreq;
function createXmlReq(){
if( window.XMLHttpRequest )
{//Mozilla、Netscape、Safari 浏览器
 xmlhttp_request = new XMLHttpRequest();
}
else
if( window.ActiveXObject )
{ //IE 浏览器
try
{ xmlhttp_request = new ActiveXObject( "Msxml2.XMLHTTP");
}
catch (e)
{
```



```
 try
{
     xmlhttp_request = new ActiveXObject( "Microsoft.XMLHTTP" ); }
}
catch(e)
{
}
}
}
}
}
```

2. Ajax 应用程序的优势

采用 Ajax 技术开发的应用程序具有下列优势：

- 通过异步模型，提升了用户体验。
- 优化了浏览器和服务器之间的传输，减少不必要的数据往返，减少了带宽占用。
- Ajax 引擎在客户端运行，承担了一部分本来由服务器承担的工作，从而减少了大用户量下的服务器负载。

异步的意思就是组件在后台工作期间，浏览器与用户保持在交互状态，并不更新当前窗口。XMLHttpRequest 对象的异步模型使用 XMLHttpRequest 对象时必须用 onreadystatechange 事件调用该对象。在触发该事件后，必须在应用程序采取行动之前检查 readyState 属性的内容。

3. 异步模型

Ajax 采用异步方法向服务器提交需求，实现 Ajax 的步骤如下：

（1）创建 XMLHttpRequest 对象。利用“xmlhttp_request=new XMLHttpRequest();”或者“xmlhttp_request = new ActiveXObject("Microsoft.XMLHTTP");”创建 XMLHttpRequest 对象。

（2）创建 HTTP 请求。创建 XMLHttpRequest 对象之后，必须为 XMLHttpRequest 对象创建 HTTP 请求，用于说明 XMLHttpRequest 对象要从何处获得数据。一般从网站或从本地获得数据。通过调用 XMLHttpRequest 对象的 Open()方法创建 HTTP 请求。其格式为：

xmlhttp_request.open('GET', URL, true);

其中，第 1 个参数是 HTTP 请求方式（GET 或 POST），按照 HTTP 规范，该参数要大写，否则，某些浏览器（如 Firefox）可能无法处理请求；第 2 个参数是请求页面的 URL；第 3 个参数用于设置请求是否为异步模式，如果是 true，JavaScript 函数将继续执行，而不等待服务器响应。

（3）设置 HTTP 请求状态变化的函数。该函数用来检查 readyState 属性，看数据是否准备就绪。如果没有准备好，隔一段时间再次检查。因为数据没有下载完时，无法使用它的属性和方法；如果已经准备好，就继续往下执行。XMLHttpRequest 对象的 onreadystatechange 属性可以设置为要使用的 JavaScript 函数名，格式为：

xmlhttp_request.onreadystatechange =FunctionName;

FunctionName 是用 JavaScript 创建的函数名，注意不要写成 FunctionName()；当然也可以直接将 JavaScript 代码创建在 onreadystatechange 之后，例如：

```
xmlhttp_request.onreadystatechange = function(){
if (http_request.readyState == 4) { //收到完整的服务器响应 }
else { //没有收到完整的服务器响应 }
};
```

首先要检查请求的状态。只有当一个完整的服务器响应已经收到，函数才可以处理该响应。XMLHttpRequest 提供了 readyState 属性来对服务器响应进行判断。readyState 属性值的具体含义如下。

- 0：表示对象已经创建，但还未初始化，即还没调用 open 方法。
- 1：表示对象已经创建并初始化，但还未调用 send 方法。
- 2：表示已经调用 send 方法，但该对象正在等待状态码和头的返回。
- 3：表示已经接收了部分数据，但还不能使用该对象的属性和方法，因为状态和响应头不完整。
- 4：表示所有数据接收完毕。

（4）发送 HTTP 请求。创建 HTTP 请求，并设置相关属性之后，就可以将 HTTP 请求发送到服务器上。使用 XMLHttpRequest 对象的 send()发送到服务器上，其格式为：

xmlhttp_request.send(null);

（5）设置获取服务器返回数据的语句。确定数据已下载完毕后，可以用 XMLHttpRequest 对象的 responseText 或 responseXML 属性来取回数据。responseText 属性是从 HTTP 响应中重新取回数据的最常见、最简单的方法。使用 responseText 属性获取返回数据可以通过函数实现，其代码如下：

```
function processRequest()
    {
  if (xmlhttp_request.readyState == 4)
  { //判断对象状态
    if (xmlhttp_request .status == 200)  //正常返回信息，状态编号 200
    { //信息已经成功返回，开始处理信息
        alert(xmlhttp_request .responseText);
         }
         else
         { //页面不正常
           alert("您所请求的页面有异常。");
         }
       }
    }
```

如果想要将 XML 文档回传给客户端，responseXML 属性是一个不错的选择，它将响应视作 XML 文档对象，然后用 DOM 迭代不同的元素、属性和文本节点。XMLHttpRequest



对象所具有的属性和方法如表 5-11 和表 5-12 所示。

表 5-11 XMLHttpRequest 对象的属性

属　性	说　明
onreadystatechange	返回或设置异步请求的事件处理程序
readyState	返回状态码：0-未初始化；1-打开；2-发送；3-正在接收；4-已加载
ResponseBody（仅 IE7）	使用无符号字节数组返回 HTTP 响应
responseText	使用字符串返回 HTTP 响应
responseXML	使用 XML DOM 对象返回 HTTP 响应
status	返回 HTTP 状态码，200 表示正常返回信息
statusText	返回描述特定 HTTP 状态码含义的文本

表 5-12 XMLHttpRequest 对象的方法

方　法	说　明
Abort	取消请求
getAllResponseHeaders	获取 HTTP 响应头的整个列表
getResponseHeader	仅获取指定的 HTTP 响应头
Open	需要使用多个参数，第 1 个设置方法属性，第 2 个设置目标 URL，第 3 个指定是同步（false）还是异步（true）发送请求
Send	发送请求到服务器
setRequestHeader	添加自定义 HTTP 头到请求

Ajax 其实是一项复杂的技术，涉及的东西很多，除了 XMLHttpRequest 对象和 JavaScript 外，还有 DOM（文档对象模型）、XML 等。JavaScript 是一个粘合剂，它通过 XMLHttpRequest 对象对浏览器端页面的诸多元素进行操控，实现与 Web 服务器的后台交互，实现数据验证、存取等功能。

5.7.2 Ajax 技术应用

下面通过 Ajax 编程示例，介绍用 Ajax 技术实现数据验证。

例 5.27 用户名唯一性数据验证。

（1）客户端文件 client.htm

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>Ajax 客户端</title>
<script language="javascript">
var xmlhttp_request = false;
 //开始初始化 XMLHttpRequest 对象
 //这段代码考虑到了 xmlhttp_request 对象与目前主流浏览器的兼容
```

```
//如果在 IE 浏览器下测试，一条
// xmlhttp_request = new ActiveXObject("Msxml2.XMLHTTP")
//或 xmlhttp_request = new ActiveXObject("Microsoft.XMLHTTP")语句即可
if(window.XMLHttpRequest)
{
//Mozilla 浏览器
xmlhttp_request = new XMLHttpRequest();
if (xmlhttp_request .overrideMimeType)
{//设置 MiME 类别
xmlhttp_request .overrideMimeType('text/xml');
}
}
else
if (window.ActiveXObject)
{
//IE 浏览器
try
{  xmlhttp_request = new ActiveXObject("Msxml2.XMLHTTP"); }
  catch (e)
{
try
   { xmlhttp_request = new ActiveXObject("Microsoft.XMLHTTP"); }
      catch (e)
       { }
      }
     }
     function send_request(url, data)
     {
      //初始化、指定处理函数、发送请求的函数
      if (!xmlhttp_request )
      { //异常，创建对象实例失败
       window.alert("不能创建 XMLHttpRequest 对象实例.");
       return false;
      }
  //确定发送请求的方式、URL 以及是否同步执行下段代码
    xmlhttp_request.open("POST", url, true);
    xmlhttp_request .onreadystatechange = processRequest;  //根据 Web 服务器应答，触发
//该状态改变事件
xmlhttp_request.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
    xmlhttp_request .send("username=" + data);  //发送信息到后台程序
     }
     ////状态改变事件处理函数：处理返回的信息
     function processRequest()
     {
  if (xmlhttp_request .readyState == 4)
  { //判断对象状态
```



```
if (xmlhttp_request .status == 200)  //正常返回信息，状态编号 200
{ //信息已经成功返回，开始处理信息
    alert(xmlhttp_request .responseText);
     }
     else
     { //页面不正常
       alert("您所请求的页面有异常。");
     }
   }
 }
 function userCheck()
 {
  var f = document.form1;
  var username = f.username.value;
  if(username=="")
  {
   window.alert("用户名不能为空。");
   f.username.focus();
   return false;
  }
  else
  {
     //该语句由用户单击“唯一性检查”按钮后执行
    send_request(" server.asp", username);
   }
 }
</script>
</head>
<body>
  <form name="form1" action="" method="post">
  用户名：<input type="text" name="username" value="">
  <input type="button" name="check" value="唯一性检查" onClick="userCheck()">
  <input type="submit" name="submit" value="提交">
  </form>
</body>
</html>
```

（2）Web 服务器端文件 server.asp

```
<%
dim username
username = request("username")
if username="zhanghua" then
 response.write("用户名" & username & "已经被注册，请更换一个用户名")
else
 response.write("用户名" & username & "尚未被使用，您可以继续")
```

```
end if
%>
```

5.8 小　结

本节首先介绍了 JavaScript 客户端脚本语言的基础知识，内容包括标识符、关键字、数据类型、常量、变量、运算符、表达式、内置对象、函数、事件；然后重点对控制语句、浏览器对象、文档对象进行了详细的分析，并列举了一些实用的例子说明客户端脚本程序的设计方法；最后对 Ajax 技术进行了简要的叙述。通过本章学习，使读者初步掌握了用 JavaScript 语言开发 Web 客户端动态网页的相关技术，为 Web 应用系统的整体开发打下了基础。

5.9 思 考 题

1．VBScript 是客户端脚本语言吗？试对其进行说明。
2．JavaScript 语言提供哪些数据类型？
3．举例说明 JavaScript 内置对象中 Array 对象的 sort()方法的用途。
4．JavaScript 提供了几种循环语句？
5．浏览器对象模型 BOM 包括哪些对象？
6．简述 DOM 模型与 BOM 模型之间的关系。
7．Ajax 技术是一种新的技术吗？简要说明它包括哪些技术？
8．利用 Ajax 技术编写简单的数据验证程序。

第6章　基于ASP.NET的服务器端程序设计

ASP.NET 是 Microsoft 公司推出的一款以.NET Framework 为基础平台的动态 Web 开发技术。Microsoft 公司将传统的 ASP（Active Server Pages）动态 Web 开发技术与.NET Framework 相结合，使得开发人员能够快速、高效、便捷地开发出可靠的 Web 应用程序。与此同时，ASP.NET 并不是 ASP 的简单升级，它使用.NET Framework 中的各种服务器端编译型语言，如 C#、VB.NET 等，并支持 Web Form、.NET 服务器控件、ADO.NET、LINQ 等高级特性。随着 Microsoft 公司不断地对.NET 平台进行扩充和完善，如今的 ASP.NET 已成为 Web 开发领域最重要的服务器端平台技术之一。

本章紧紧围绕开发一个简单的 Web 应用程序所需要的基础知识展开讨论，希望为读者进一步学习和掌握 ASP.NET 应用程序开发起到抛砖引玉的作用。首先对.NET Framework 及 ASP.NET 做简要介绍，然后分别阐述 Web 应用程序开发相关的基础知识，包括 CSS 工具、服务器控件、数据访问、主题与外观、母版页和内置对象等，这些内容在使用 ASP.NET 技术开发 Web 应用程序时都是至关重要的。

6.1　ASP.NET 简介

ASP.NET 是一个统一的 Web 开发模型，它包括多种服务以便于开发人员使用尽可能少的代码生成企业级 Web 应用程序。开发人员可以使用与公共语言运行库（Common Language Runtime，CLR）兼容的任何语言（包括 Visual Basic、C#、JScript.NET 和 J#）来编写代码访问.NET Framework 中的类，开发利用公共语言运行库、类型安全、继承等优点的 ASP.NET 应用程序。

6.1.1　.NET 框架概述

.NET 代表了一个集合、一个环境、一个可以作为平台支持下一代 Internet 的可编程结构，它通过使用 HTTP、XML 等标准，使得各个系统平台间的互操作成为现实。.NET 的最主要部分是.NET 框架，.NET 框架设计为一个集成环境，可以在 Internet、桌面（如 Windows 窗体）甚至移动设备（使用精简框架 Compact Framework）上无缝地开发和运行应用程序。从概念上讲，.NET 框架代表了一种崭新的软件开发模式，它与 Win32 API 或 COM 一样，是把系统服务以接口形式提供给开发人员的软件开发平台。与以往不同的是，.NET 框架能够更好地完成代码重用、资源配置、多语言集成开发和安全管理等任务，在安全性、易用



性以及开发效率等方面远远超过了以往的开发模式。

.NET 框架也被理解为一种新的计算平台，它简化了在高度分布式 Internet 环境中的应用程序开发，支持超过 20 种不同的编程语言，帮助开发人员把精力集中在实现商业逻辑的核心上，使开发人员的经验在面对不同类型和规模的应用程序（如基于 Windows 的应用程序和基于 Web 的应用程序）时保持一致，在未来的版本中甚至仅在程序发布时才需要指定发布的类型（作为 WinForm 还是 WebForm）。

如图 6-1 所示，.NET 框架由两部分组成：通用语言运行时（CLR，.NET 开发人员的源代码和硬件底层之间的中间媒介，所有的.NET 代码都在 CLR 中运行）和框架类库（FCL，包括数据访问组件、基础类库以及 WebForm、WinForm、Web Services 模板）。

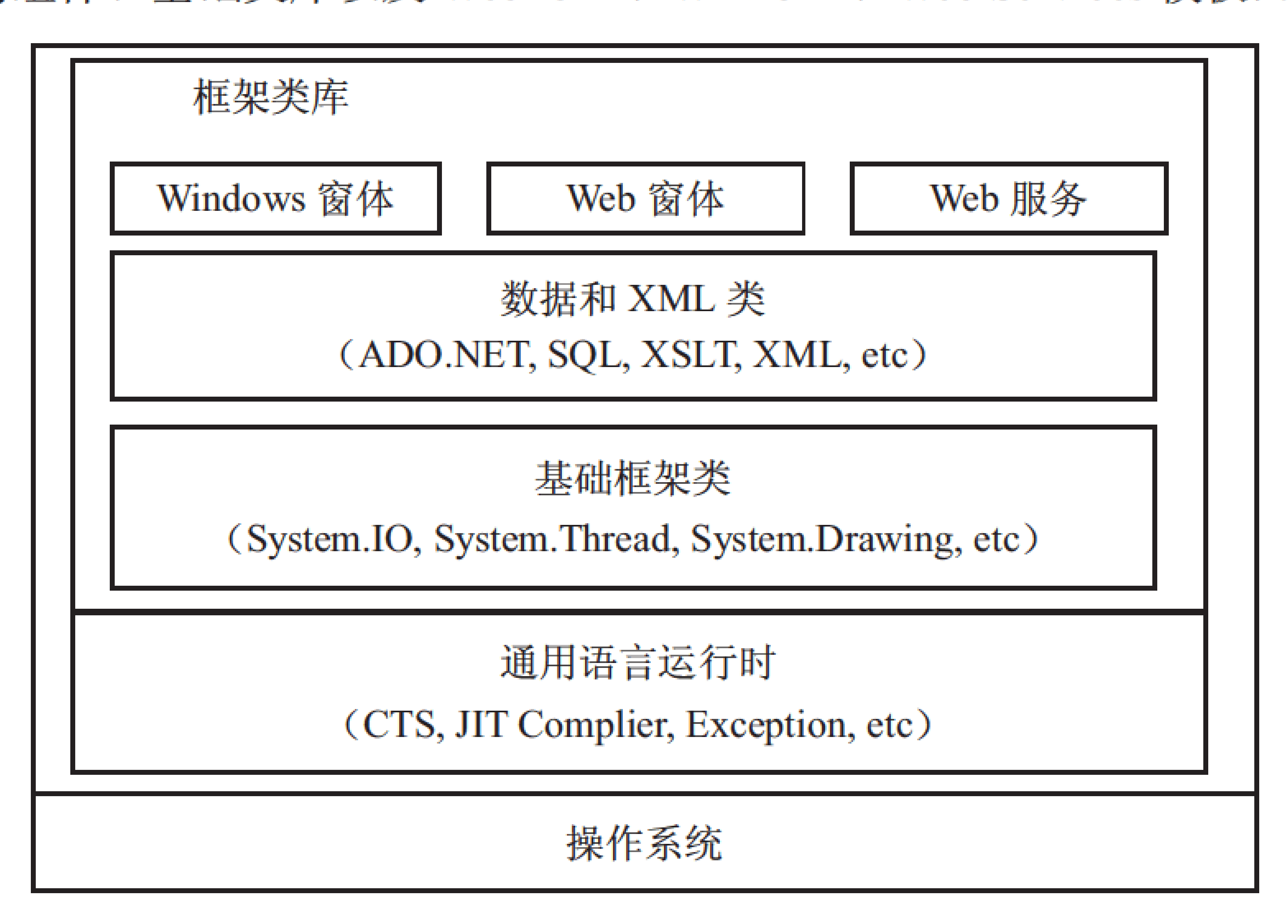


图 6-1　.NET 框架的组成

自从微软推出第 1 版的.NET Framework，已经经过了多个版本的演变，目前发布的最新版本是.NET Framework 3.5。下面列出了.NET 技术平台的发展过程。

- 2000 年 6 月，Microsoft 公司提出了.NET 的构想。
- 2002 年 1 月，Microsoft 公司正式发布了.NET Framework 1.0 正式版和 Visual Studio 2002 版本。
- 2003 年 4 月，Microsoft 公司推出了.NET Framework 的 1.1 版本和 Visual Studio 2003。
- 2004 年 6 月，Microsoft 公司发布了.NET Framework 2.0 的 Beta 1 版本和 Visual Studio 2005 Beta1 版本。
- 2005 年 4 月，Microsoft 公司发布 Visual Studio 2005 Beta2 版本。
- 2005 年 11 月，Microsoft 公司发布 Visual Studio 2005 和 SQL Server 2005 正式版。
- 2006 年底，Microsoft 公司又发布了.NET Framework 3.0 版本。
- 2007 年 11 月，Microsoft 公司发布了最新的.NET Framework 3.5 和 Visual Studio 2008 版本。
- .NET Framework 4.0 和 Visual Studio Team System 2010 将会在 2010 年发布。

本章接下来的内容是基于.NET Framework 3.5 展开的。



6.1.2　开发环境简介

尽管从理论上讲，只用 Notepad 或其他文本编辑器就可以写出 ASP.NET 应用程序，但最好还是安装一个 Microsoft Visual Studio 的副本，因为其中包含了大量有助于快速创建复杂 ASP.NET Web 应用程序的工具和一套在基本的代码管理之上的高级特性，致力于提高开发效率，将开发人员从繁重、繁琐的开发任务中解放出来。下面列出了 VS 2008 中的一些优点和新特性：

- 集成的 Web 服务器。运行 ASP.NET Web 应用程序需要 Web 服务器软件（如 IIS），它等待 Web 请求并处理适当的页面。安装 Web 服务器并不难，但也不方便。由于 VS 2008 内集成了用于开发的 Web 服务器，开发人员可以在无须安装 IIS 的情况下，从设计环境直接运行网站。
- 多语言开发。VS 2008 允许开发人员在任何时候在同一个接口（IDE）下使用某种语言或者多种其他语言来编程。不仅如此，VS 2008 还允许开发人员用不同的语言构建 Web 页，但是要把它们包括在一个 Web 应用程序中，唯一的限制就是不能在同一个网页中使用两种以上的语言（这样会导致明显的编译问题）。
- 更少的代码。大多数应用程序都需要一些标准模板文件代码，ASP.NET 网站也不例外。例如，当向一个网页添加新的控件、附加事件处理程序或调整格式时，需要在页面标记中设置许多细节，这些基本的任务都由 VS 2008 自动完成。
- 直观的编码风格。默认情况下，VS 2008 会自动格式化代码并且使用不同的颜色来标识各种元素，如注释，这些使代码更具有可读性而且少出错。
- 多目标支持。Web 服务器不会一夜之间全部从.NET 2.0 迁移到.NET 3.5，开发人员可以根据目标环境，利用 Visual Studio 自由开发自 2.0 版本开始的各种.NET Framework 版本的应用程序。
- CSS 工具。为了保持整个网站的一致性，开发人员通常会使用 CSS 标准。现在，Visual Studio 使得 Web 页面和样式表的关联更加简单，不必手工编辑标记，只要选择要应用到页面各种元素的样式即可。

Visual Studio 2008 有多个版本，如 Visual Studio 2008 专业版本和 Visual Studio 2008 Team System。如果计算机资源受限，或出于版权的考虑，开发人员也可以使用 Visual Web Developer 2008 Express 的简化版本，当然功能上会相对少一些，但是它包含了创建复杂且功能丰富的 Web 应用程序所需的所有功能和工具。

本章所有例子均在免费版 Visual Web Development 2008 Express Edition（简称 VWD 2008，可以从微软官方网站上下载）上运行和测试，也适用于正式版的 Visual Studio 2008 产品。

6.1.3　创建第一个 ASP.NET 网站

VWD 2008 提供了如下两种创建 ASP.NET Web 应用程序的方法：

- 基于项目的开发。创建一个 Web 项目时，VWD 2008 生成一个.csproj 项目文件（假设使用 C#编码），它记录项目中的文件并保存一些调试设置。运行 Web 项目时，VWD 2008 在启动 Web 浏览器前把项目的所有代码编译成一个程序集。
- 无项目文件的开发。另一个可选的方式是创建一个没有任何项目文件的简单网站。此时，VWD 2008 认为在网站目录（及其子目录）中的所有文件都是 Web 应用程序的一部分。在这样的场景里，VWD 2008 不需要预编译代码，而是由 ASP.NET 在第一次请求页面时编译网站。

下面开始创建标准的无项目文件的网站，它是更简单、更直接的方式。首先启动 VWD 2008，如图 6-2 所示。要想立刻着手创建一个新的 Web 应用程序，可选择“文件”→“新建网站”命令或在图 6-2 所示的起始页中选择“创建”后的“网站”选项，将打开“新建网站”对话框，如图 6-3 所示。



图 6-2　VWD 2008 主界面

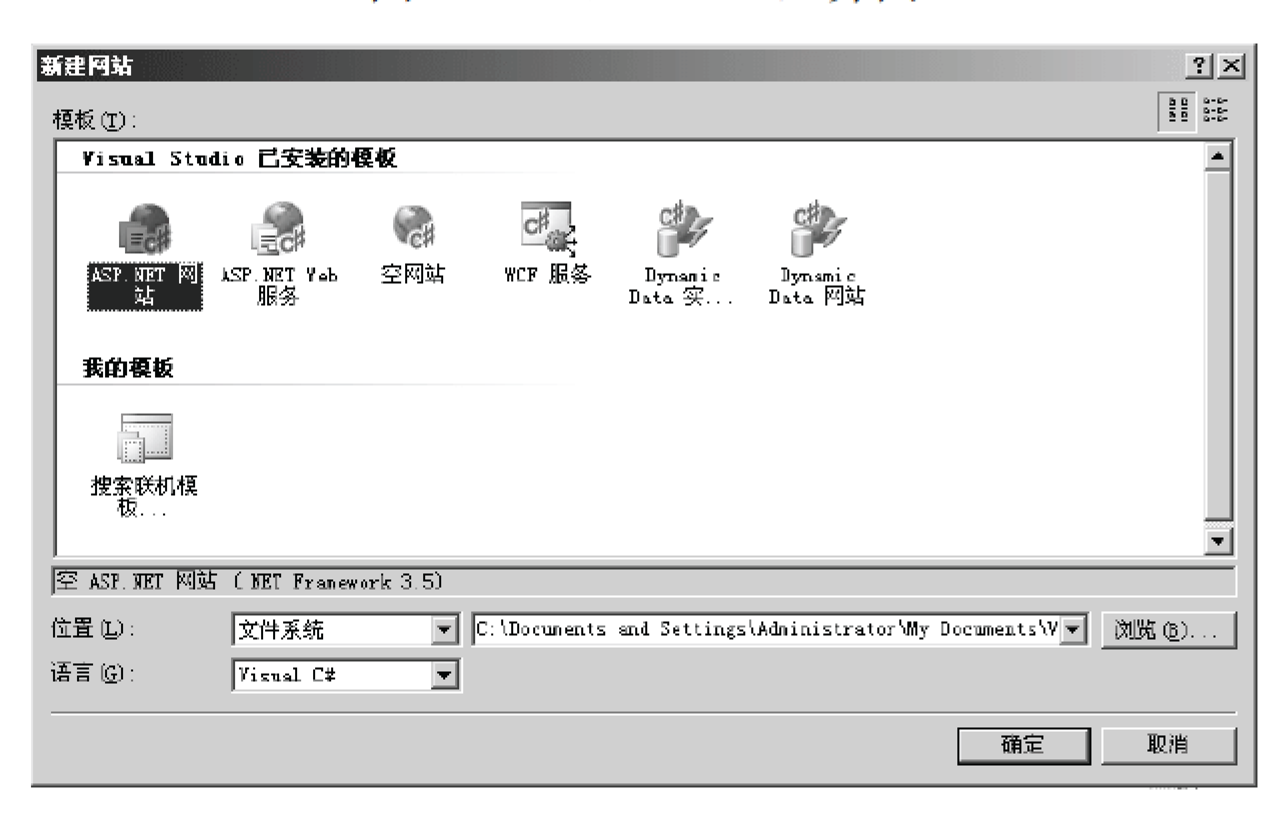


图 6-3　新建 Web 站点

在“新建网站”对话框中可以提供如下 3 个细节：

- 模板。模板决定网站以何种文件作为开始。VWD 2008 支持两种类型的 ASP.NET 基本应用程序，分别为网站应用程序以及 Web 服务应用程序。这些应用程序实际上以同样的方式编译和执行。可以将一个网页添加到一个 Web 服务应用程序中，



也能把一个 Web 服务添加到一个普通的 Web 应用程序中。唯一的不同之处就是 VWD 2008 默认创建的文件。在 Web 应用程序中，将以一个示例页面作为项目的开始。在 Web 服务应用程序中，将以一个示例 Web 服务作为开始。另外，VWD 2008 包含许多的模板以便适应不同类型的网站，并且开发人员也可以创建自己的模板（或者下载第三方模板）。

- 位置。位置指定了网站文件存储的地方。一般情况下，开发人员会选择“文件系统”并指定本机或者网络路径上的一个文件夹，也可以通过 HTTP 或者 FTP 来编辑一个网站目录。
- 语言。语言用来指定使用何种.NET 编程语言来编写网站。所选择的语言就是项目的默认语言。这也就意味着可以明确地将一个 Visual Basic 网页添加到一个 C#网站中，反之亦然（这一点对于 Visual Studio 的老版本来说是不可能的）。

单击“确定”按钮后，就会出现如图 6-4 所示的界面，这就表示已经创建了一个最简单的网站。在讨论进一步的内容之前，需要对 VWD 界面的主要组成部分做进一步的了解。图 6-4 标出了 VWD 窗口的最常用部分，表 6-1 针对每个部分都做了相应的描述。

主菜单　工具箱　文档窗口　起始页　解决方案资源管理器

错误列表　属性面板

管理样式　单击此处，切换到“设计”视图

图 6-4　VWD 2008 主界面

表 6-1　VWD 窗口介绍

窗　口	描　述
主菜单	包括 Windows 应用程序中的常规菜单，如 File、Edit、Help 等，以及一些 VWD 特有的菜单，根据执行的具体任务，这个菜单也会有很大的变化
解决方案资源管理器	将 Web 应用程序中的文件和子文件夹列出，可以通过右击来添加文件和文件夹
工具箱	显示 ASP.NET 自带的服务器控件、加入到工具箱的第三方控件及自己开发的用户自定义控件。可以使用任何语言来编写并且可以被任何语言来调用
起始页	可以快速地创建新的 Web 站点或者打开现有站点，还可以用 Start Page 来访问一些常见的帮助主题并显示 Microsoft Web 站点中的标题
属性面板	允许开发人员来配置当前所选择的元素，可以是解决方案资源管理器中的一个文件或者一个 Web 表单设计界面上的控件
错误列表	报告 Visual Studio 在代码中检测所发现的、尚未解决的错误
文档窗口	允许开发人员在解决方案资源管理器中通过拖放来设计网页以及编辑代码文件。同样支持非 ASP.NET 文件类型，如静态 HTML 以及 XML 文件
管理样式和应用样式	允许开发人员在链接样式表中修改样式并把它们应用到当前网页
视图切换按钮	有设计、分拆和源 3 种视图，分别对应对一个 Web 页面的不同观察方式

选择“调试”→“开始执行（不调试）”命令，将出现如图 6-5 所示的运行结果。整个工作不应该到此为止，而只是刚刚开始，因为这个网站没有任何功能，更谈不上美观，接下来将从几个方面介绍如何完善这个网站。

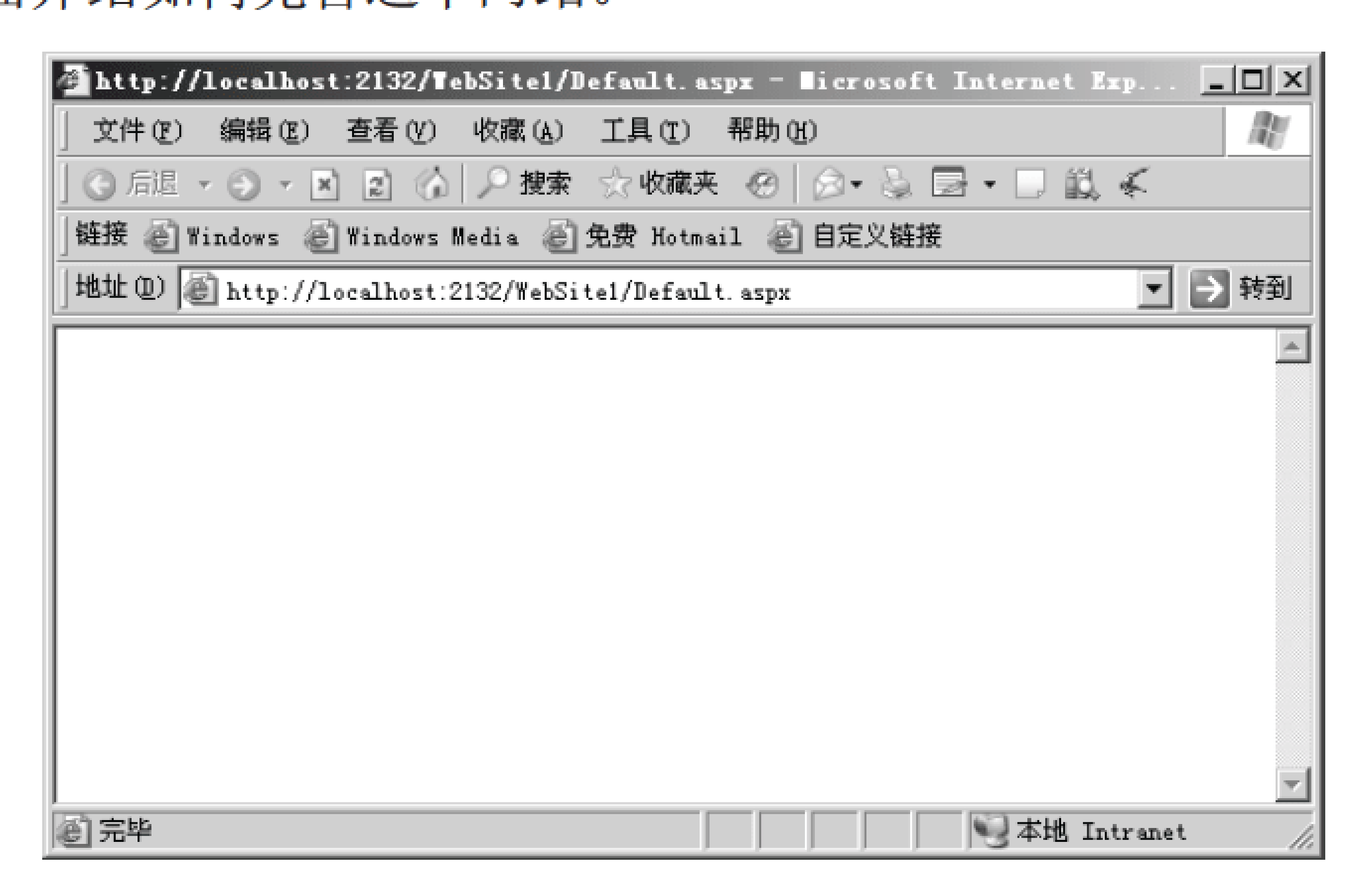


图 6-5　空网站运行示意图

6.1.4　应用需求简介

由于 ASP.NET 内容及其复杂，为了有针对性地学习并使用它，本章后续内容（第 6.2 节除外）将主要围绕一个简单的应用需求展开，即实现一个基于 ASP.NET 的通讯录管理系统，具备通讯录信息（如姓名、联系电话、E-mail 和分组类型等）的查看和维护功能。



6.2　在 VWD 2008 中进行 HTML 和 CSS 设计

关于 HTML 和 CSS 的基础知识前面章节已经叙述，本章主要介绍在 VWD 2008 开发环境下如何使用 HTML 工具和 CSS 工具设计 Web 页面。

首先新建一个网站或打开现有网站，如果需要在新页面上练习，则可以在解决方案资源管理器上右击，然后在弹出的快捷菜单中选择“添加新项”命令，打开如图 6-6 所示的“添加新项”对话框，在其中选择“Web 窗体”选项并单击“添加”按钮，添加一个 Web 文件。

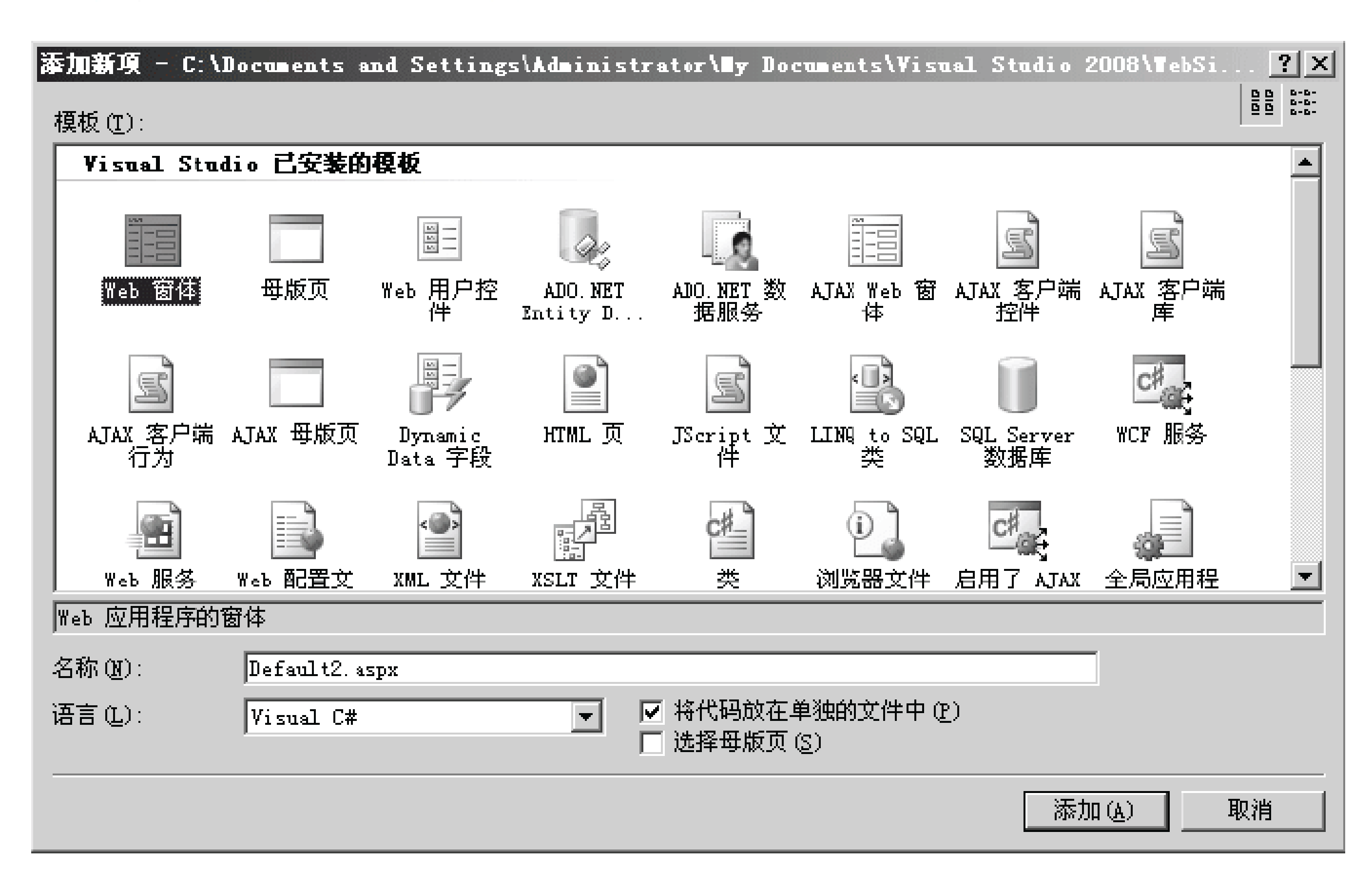


图 6-6　添加新项

Web 文件是 Web 应用程序特有的文件，从图 6-6 中也能看到很多其他类型的文件，这些文件分别有各自的用途。表 6-2 列出了各种 Web 文件及其扩展名，并说明了各种文件的用法。

表 6-2　ASP.NET 文件类型

文 件 类 型	扩　展　名	说　　明
Web Form 与 Ajax Web Form	.aspx	这些文件是所有 ASP.NET Web 站点都要用到的文件。Web Form 是用户在浏览器中浏览的页面
Master Page 与 Ajax Master Page	.master	这些文件允许开发人员定义 Web 站点的全局结构和外观。在第 6.5.2 节可以看到它们的用法
Web User Control	.ascx	含有可重复用在站点的多个页面中的页面片段
HTML Page	.htm/.html	可用来显示 Web 站点中的静态 HTML
Style Sheet	.css	含有允许开发人员定制 Web 站点的样式和格式的 CSS 代码。第 6.2.2 节将介绍关于 CSS 的更多信息
Web Configuration File	.config	含有用在整个站点中的全局配置信息。在本书后面会了解如何使用 web.config
Site Map	.sitemap	含有一个层次结构，表示站点中 XML 格式的文件，Site Map 用于导航

续表

文 件 类 型	扩 展 名	说 明
JScript File	.js	含有可以在客户机的浏览器中执行的 JavaScript(Microsoft 称之为 JScript)
Skin File	.skin	含有设计 Web 站点中的控件的信息。Skin 将在第 6.5.1 节介绍

6.2.1 使用 HTML 工具设计页面

在 VWD 2008 中向页面添加 HTML 有多种方式，可以简单地在“源”视图中输入（详见第 2 章的相关内容），但并不推荐这样做，因为它会强制手工输入大量代码，且容易出错。除此之外，VWD 2008 提供了几个有用的工具，能够帮助开发人员更轻松地向页面中插入新 HTML 并向其应用格式，这些工具包括“格式”（Formatting）工具栏、“格式”（Format）菜单和“表”（Table）菜单。

提示： 要使这些工具变为活动的，需要让文档在设计视图中。如果是在分拆视图模式下工作，则一定要确保焦点在设计视图，否则就会发现大多数工具都不可用。

1. 插入和格式化文本

在 Web 页面的“分拆”视图和“源”视图中都可以输入文本，只要将光标放在所需的位置并开始输入即可。当切换到“设计”视图时，“格式”工具栏变得可用，其中的选项如图 6-7 所示。

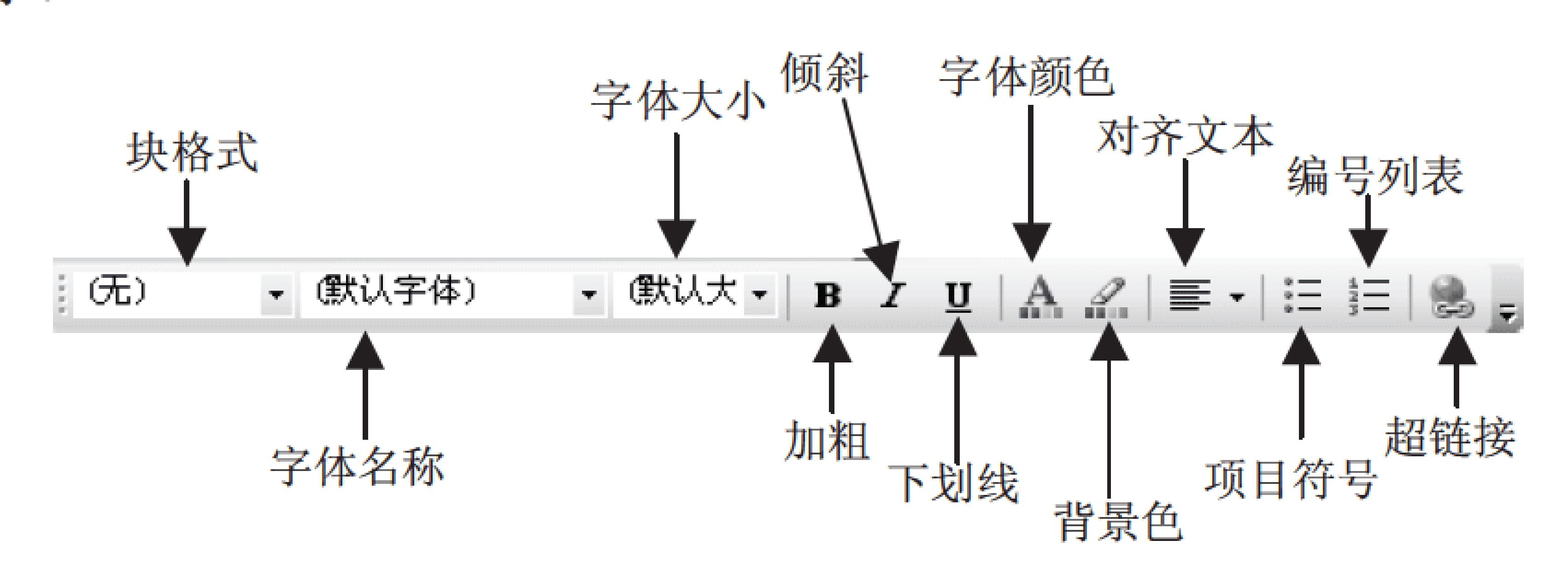


图 6-7 “格式”工具栏

“块格式”下拉列表框可以插入 HTML 标记，如<p>表示段落，<h1>～<h6>表示标题，<ul>、<ol>、<li>标记表示列表，可以直接从该下拉列表框中选择一项插入页面中，或者也可以先选择一些文本，然后从列表中选择适当的块元素，并把选中的文本包在标记内。“字体名称”下拉列表框允许修改字体，“字体大小”下拉列表框可以用来修改字体的大小。该工具栏上的其余按钮的功能与其他编辑环境（如 MS Word）中的相应功能完全相同。例如，B 按钮用粗体字格式化文本。类似地，I 和 U 按钮分别使字体倾斜和加下划线。

例 6.1 利用上面介绍的工具，就可以在网页中添加带有格式的文本，页面设计视图如图 6-8 所示。





图 6-8　HTML 格式化页面

对应的页面标记程序如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default2.aspx.cs" Inherits="Default2" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <h1 align="center">
    <strong><font color="#ff0000"><span style="font-size: x-large;font-family: 黑体;">沁园春·雪
</span> </font></strong>
        </h1>
        <p align="center">
            <a href="http://">毛泽东 </a>     一九三六年二月
        </p>
        <p  style="line-height: 150%;text-align: center;width: 720px;margin-top: 0;margin-bottom: 0;">
         <span  style="font-family: 幼圆;font-size: x-large;">北国风光</span>，<br />
            千里冰封，<br />
          <span  style=" font-size: large;font-style: italic;">万里雪飘。</span><br />
            望<b>长城</b>内外，<br />
            惟余莽莽；<br />
            <b>大河</b>上下，<br />
            顿失滔滔。<br />
         <span  style=" font-size: smaller;font-weight: bold;">山舞银蛇，</span><br />
            原驰蜡象，<br />
            欲与天公试比高。<br />
            须晴日，<br />
            看<span  style=" color: #FF0000; font-size: xx-large;font-weight: bold;">红装</span>
素裹，<br />
            分外妖娆。
```




```
            </p>
        </div>
        </form>
</body>
</html>
```

2．添加表（Table）

HTML 表是用来规范网页上元素布局的非常好的途径。例如，如果页面中含有上、中、下或左、中、右三大块区域，就可以使用一个 3 行表或 3 列表来实现。按前述方法添加一个新页面，然后在“设计”视图中选择“插入表”命令，弹出如图 6-9 所示的对话框，可以设置表格的大小、布局、边框和背景等属性。

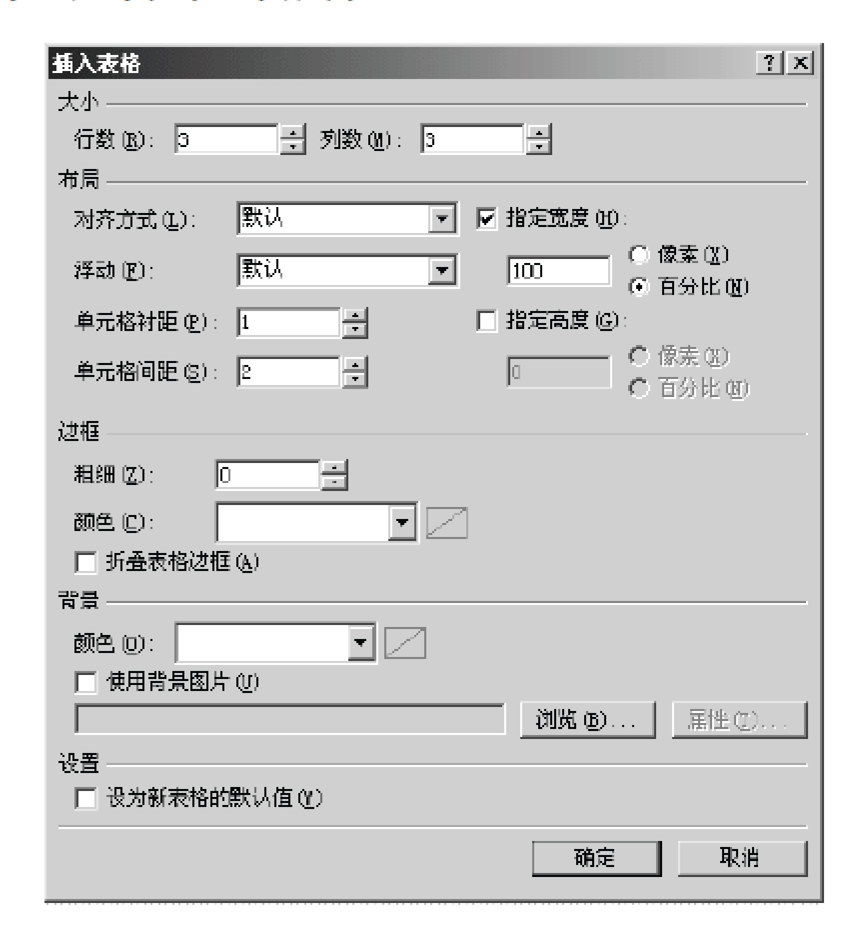


图 6-9　插入 HTML 表格

单击“确定”按钮，打开如图 6-10 所示的界面。

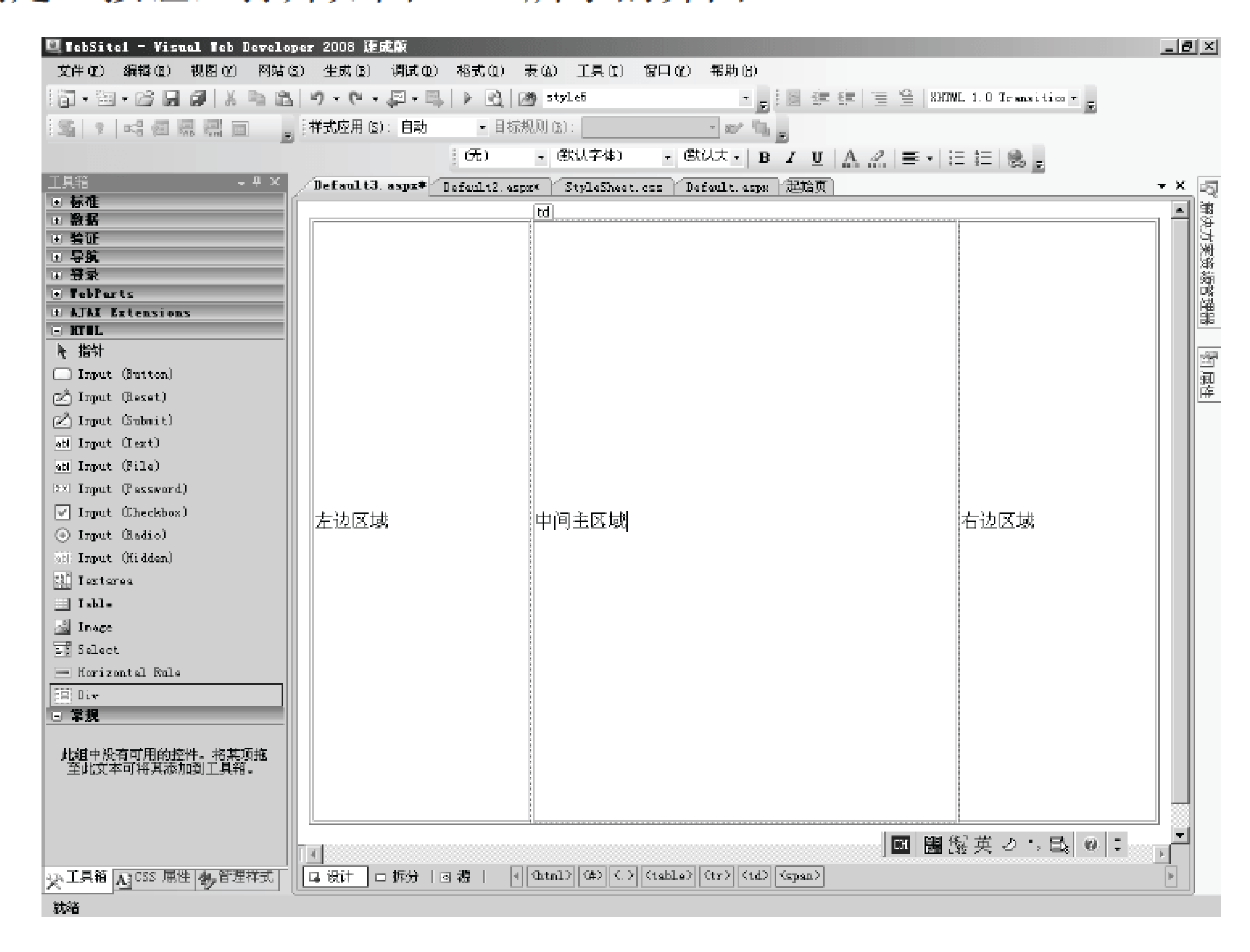


图 6-10　HTML 表格设计视图



对应的页面标记程序如下：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
<form id="form1" runat="server">
    <div>
     <table border="1"  style=" width: 100%;border-style: solid;border-width: 1px;">
            <tr>
                <td  style="height: 521px;width: 185px;">
                    <span lang="zh-cn">左边区域</span></td>
                <td  style=" height: 521px;width: 365px;">
                    <span lang="zh-cn">中间主区域</span></td>
                <td  style=" height: 521px;">
                    <span lang="zh-cn">右边区域</span></td>
            </tr>
        </table>  </div>
    </form>
</body>   </html>
```

延伸阅读提示：DIV+CSS 是流行的网页布局方式，一本不错的参考书是《变幻之美——Div+CSS 网页布局揭秘（案例实战篇)》(金峰编著，人民邮电出版社)。

6.2.2　使用 CSS 工具设计页面

CSS 是给 Web 站点添加样式的一种极佳方式，它是一种强大而方便的机制，允许开发人员为 Web 站点创建复杂的样式和布局，并能够解决 HTML 在格式化方面造成的多种问题，如产生大量标记、内容与格式混在一个文件中等。使用 CSS 创建漂亮的 Web 站点设计也需要大量代码，很容易令人畏缩。同样，VWD 2008 也提供了许多工具，帮助简化页面布局和 CSS 管理，下面分别介绍这些工具。

- 新建样式对话框：用来可视化地创建各种样式和设置属性值。
- 管理样式对话框：用来组织站点中的样式，将它们从嵌套样式表改为外部样式表，反之亦然；对它们重新排序，将现有样式表链接到一个文档，并创建新的内联、嵌套或外部样式表。
- 应用样式对话框：用来从站点中选择所有可用样式，并将它们快速应用到页面中的不同元素上。
- 样式表工具条：用来快速创建新规则与样式。
- “CSS 属性”面板：用来修改属性值。
- “修改样式”对话框：可以用来可视化地创建声明。



- “添加样式规则”对话框：帮助开发人员构建较复杂的选择符。

1. “新建样式”对话框

使用 VWD 2008 设计界面时，很容易为 Web 页面创建新样式。从“格式”菜单中选择“新建样式（New Style）”命令，会打开如图 6-11 所示的“新建样式”对话框。



图 6-11　新建样式

- 从“选择器”下拉列表框中选择要创建的选择器类型，其中包含所有的元素类型，如果要创建类选择器或 ID 选择器，只需在“选择器”下拉列表框中输入样式名称。
- 从“定义位置”下拉列表框中选择创建样式的位置，可以选择“当前页面”创建一个内部样式表，选择“新建样式表”创建一个新的外部样式表文件，选择“现有样式表”把样式插入已有的样式表文件中。如果选择了“新建样式表”或“现有样式表”，就需要为 URL 下拉列表框提供一个值。
- 从“类别”列表框中选择属性类别，就可以开始设置样式属性值。VWD 2008 把常见 CSS 属性划分为字体、块、背景、边框、方框、定位、布局、列表、表格 9 大类别。“预览”列表框中会给出新样式的实时预览。另外，“说明”文本框中显示了 VWD 2008 创建的属性语法。

例 6.2　为下面的 CSS 规则分别添加一个外部样式表、嵌套样式和级联样式。

```
.Title
{
    font-family: 微软雅黑;
    font-size: large;
    font-weight: bold;
}
#Content
{
```



```
        font-family: 楷体_GB2312;
        font-size: x-large;
        font-weight: normal;
        font-style: italic;
    }
```

（1）如图 6-11 所示，在“定义位置”下拉列表框中选择“新建样式表”选项，在“类别”列表框中选择“字体”选项，并设置 font-family、font-size、font-weight 属性。单击“确定”按钮后，解决方案资源管理器中创建样式表文件 StyleSheet.css，内容就是刚刚添加的样式，如图 6-12 所示。

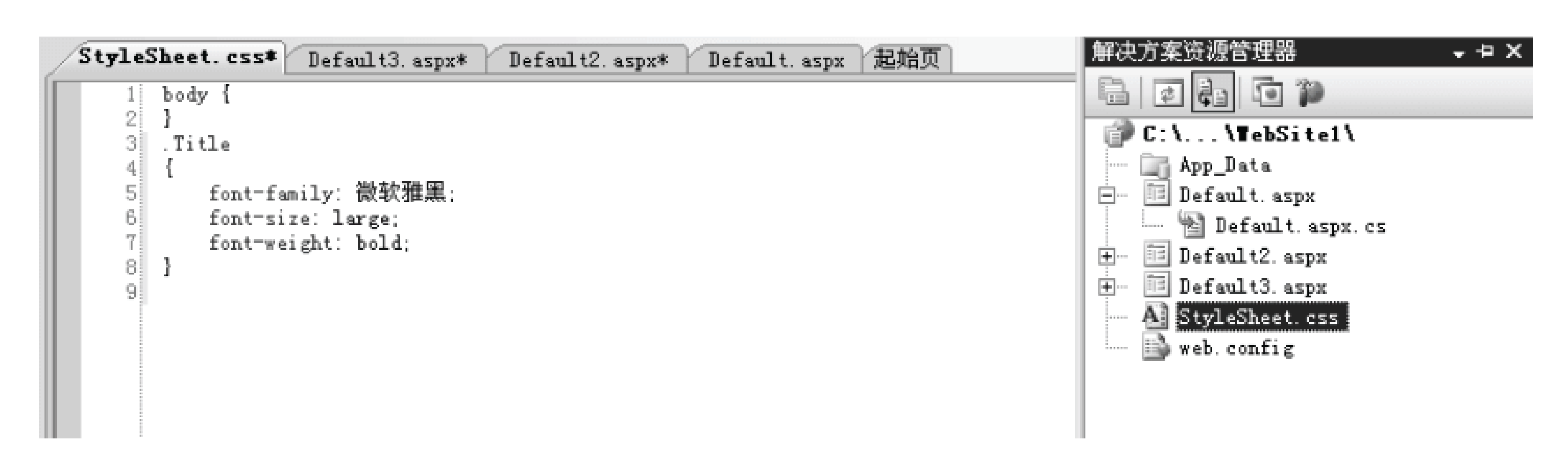


图 6-12　外部样式表

（2）在图 6-11 中的“定义位置”下拉列表框中选择“当前网页”选项，其余与步骤（1）类似。单击“确定”按钮后，相应页面<head>标记中即生成<style>标记及 CSS 规则。对应的页面标记程序如下：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        #Content
        {
            font-family: 楷体_GB2312;
            font-size: x-large;
            font-weight: normal;
            font-style: italic;
        }
    </style>
</head>
<body>
<form id="form1" runat="server">
    …
    </form>
</body>
</html>
```

（3）与前面步骤不同的是，添加级联样式时必须首先选中某个页面标记，然后打开“新建样式”对话框，在“选择器”下拉列表框中选择“级联样式”，定义位置不用选，其余与

步骤（1）一样。单击“确定”按钮后，页面标记的样式自动更新，如图 6-13 所示。

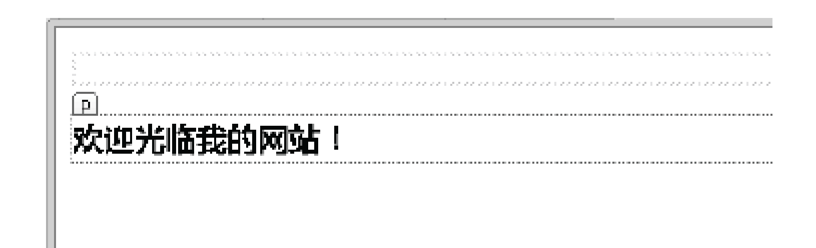


图 6-13　应用级联样式表

切换到“源”视图，会发现相应标记中增加 style 属性，程序代码如下：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server" >
    </form>
    <p style="font-family: 微软雅黑; font-size: large; font-weight: bold">
        <span lang="zh-cn">欢迎光临我的网站！</span></p>
</body>
</html>
```

2. “管理样式”对话框

在“视图”菜单中选择“管理样式”命令，即可打开“管理样式”对话框，如图 6-14 所示。

“管理样式”对话框的顶部包含两个重要的超链接：新建样式和附加样式表。单击“新建样式”超链接可以打开“新建样式”对话框，创建新的 CSS 样式，如本节前面内容所述。单击“附加样式表”超链接可以把新样式表导入 Web 页面，如图 6-15 所示。



图 6-14　“管理样式”对话框



图 6-15　附加外部样式表

单击“确定”按钮，则把样式表关联到 Web 页面上，会使 VWD 2008 在 Web 页面上插



入<link>标记，程序代码如下：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
    <link href="StyleSheet.css" rel="stylesheet" type="text/css" />
    <style type="text/css">
        #Content
        {
            font-family: 楷体_GB2312;
            font-size: x-large;
            font-weight: normal;
            font-style: italic;
        }
    </style>
</head>
<body>
…
</body></html>
```

必须小心地安排<link>标记和样式块的顺序，确保样式得到正确的应用。

“管理样式”对话框中间显示了在 VWD 2008 中打开的 Web 页面上能使用的所有样式。样式根据其选择器类型使用的彩色项目列表进行了彩色编码：蓝色表示类型选择器，绿色表示类选择器，红色表示 ID 选择器。页面中使用的样式用包围彩色项目列表的灰色圆框起来。

另外，在图 6-15 中展开 StyleSheet.css 节点，就会列出该样式表包含的所有样式。

“管理样式”对话框底部包含一个预览区域，可以查看选中样式的实时预览。

3. “应用样式”对话框

在 VWD 2008 中，“应用样式”对话框中也提供了一种非常方便的方式，可以查看应用程序中的 CSS 样式，把这些样式应用于 Web 页面上的元素。在该对话框中，可以把 CSS 文件关联到 Web 页面上，使外部的 CSS 样式可用。选择要应用的页面样式或从元素中删除的页面样式、修改样式等。与其他 CSS 工具窗口一样，“应用样式”对话框也根据 CSS 继承顺序显示可用的样式，外部样式先显示，之后是页面样式部分，最后是内嵌样式。“应用样式”对话框还可以根据当前选择的元素，显示应用程序中可以应用于该元素类型的样式。样式还可以根据 CSS 选择器样式来组合，使每个选择器类型有不同外观的指示器。

在主菜单中选择“视图”→“应用样式”命令，打开“应用样式”对话框，如图 6-16 所示，就可以轻松地向页面中的元素应用样式规则。同样，该对话框顶部也有“新建样式”和“附加样式表”超链接。

该对话框中间显示了段落标记<p>的可用样式，分别是关联文件 StyleSheet.css 中的所有样式、当前页面中的样式、指定元素上已应用的内嵌样式。单击这些样式，就会把它们应用于元素。



图 6-16　应用样式

4. “CSS 属性”面板

“CSS 属性”面板显示了已应用于当前所选元素的所有 CSS 属性，如图 6-17 所示，该面板由两部分组成：“应用的规则”栏和“CSS 属性”栏。



图 6-17　“CSS 属性”面板

“应用的规则”栏中显示了应用于所选元素的所有 CSS 规则，该列表自动排序，显示已应用规则的继承链，最外层的规则在顶部，最内层的规则在底部。这表示包含在外部 CSS 文件中的规则自动排在列表的顶部，内嵌样式排在底部。可以单击列表中的每个规则，修改显示其在“CSS 属性”栏中的属性。“CSS 属性”栏中显示了元素可用的所有 CSS 属性，其中已经设置了值的属性显示为粗体。另外，还可以在“CSS 属性”栏中直接给 CSS 规则设置属性值。在“CSS 属性”面板中还有一个“摘要”按钮，单击可以改变 CSS 属性网格的显示，只列出已设置了值的属性。

5. “修改样式”对话框

在“CSS 属性”面板、“管理样式”对话框或“应用样式”对话框中，可以右击列表中的每个规则，然后在弹出的快捷菜单中选择“修改样式”命令，即可打开如图 6-18 所示的



"修改样式"对话框（与"新建样式"对话框类似），在打开的外部样式表文件中右击文档区域，在弹出的快捷菜单中选择"生成样式"命令也会打开该对话框。



图 6-18　修改样式

与 CSS 属性工具类似，在该对话框中可以修改所选规则的任意属性值，还可以修改选择器名称。

6. "添加样式规则"对话框

使用"新建样式"对话框添加的样式选择符都是单一的，不能反映选择符之间的层次结构。可以在外部样式表文件中右击文档区域，在弹出的快捷菜单中选择"添加样式规则"命令，即可打开如图 6-19 所示的"添加样式规则"对话框，在其中构建复杂的选择符。有关复合样式规则的详细介绍，读者可参考其他相关文献。

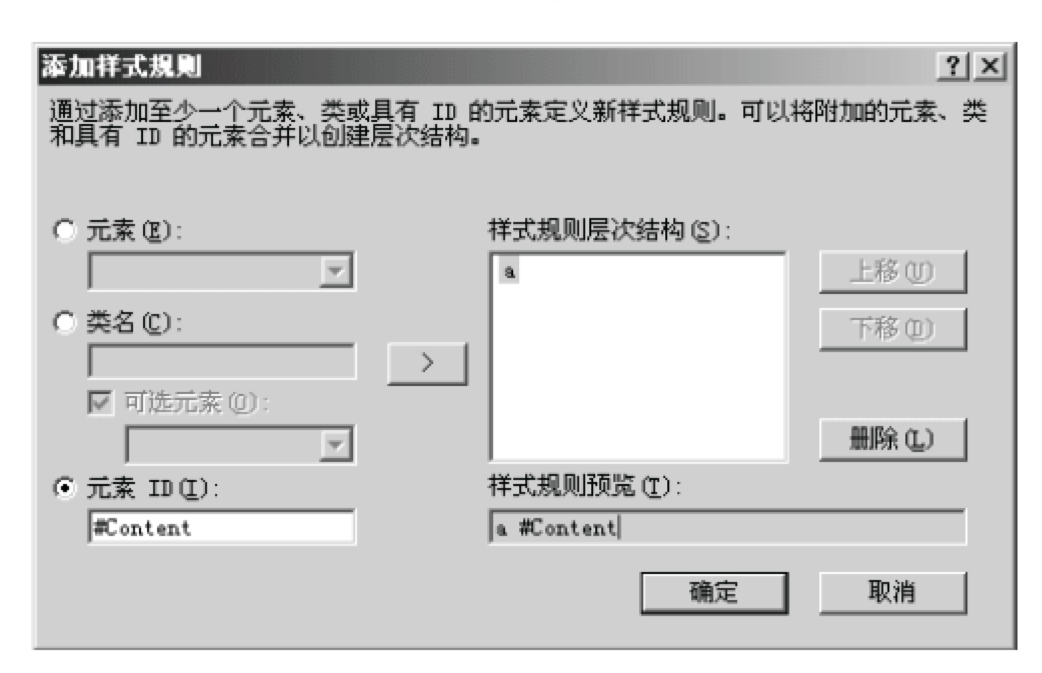


图 6-19　"添加样式规则"对话框

6.3　使用 ASP.NET 服务器控件

控件是构建图形用户界面（GUI）的模块，如果您有 Windows 窗体应用程序开发的经

历，肯定会熟悉很多控件，包括文本框、按钮、复选框和列表框等。控件为用户提供了显示偏好、输入数据或者做出选择的途径。它们也可以在一定范围内，提供基础功能方面的支持，如验证、数据操作、母版页和安全性。

6.3.1 ASP.NET 控件的类型

VWD 2008 内置了丰富的控件（如图 6-20 所示），可以帮助开发人员快速有效地创建 ASP.NET 网页。这些 Web 控件可分为如下 4 种类型。



图 6-20　VWD 2008 工具箱

1. HTML 服务器控件

对服务器公开的 HTML 元素，可对其进行编程。HTML 服务器控件公开一个对象模型，该模型十分紧密地映射到相应控件所呈现的 HTML 元素。由于篇幅所限，本章不再讨论。可以参考 O'Reilly 出版的由 Chuck Musciano 和 Bill Kennedy 所著的《HTML 和 XHTML 权威指南（第 5 版）》（HTML and XHTML: The Definitive Guide, Fifth Edition）。

2. Web 服务器控件

Web 服务器控件比 HTML 服务器控件具有更多内置功能，不仅包括窗体控件（如按钮和文本框），而且还包括特殊用途的控件（如日历、菜单和树视图控件）。Web 服务器控件与 HTML 服务器控件相比更为抽象，因为其对象模型不一定反映 HTML 语法。

3. 验证控件

验证控件可以认为是 Web 服务器控件的特殊类型，它们包含逻辑以允许对用户在输入控件（如 TextBox 控件）中输入的内容进行验证。验证控件可用于对必填字段进行检查、对照字符的特定值或模式进行测试、验证某个值是否在限定范围之内等。

4. 用户控件

用户控件是由开发人员创建的控件，ASP.NET 用户控件可以嵌入到其他 ASP.NET 网页中，这是创建工具栏和其他可重用元素的一条捷径。

其中 Web 服务器控件是 ASP.NET 控件的核心，也是本章介绍的重点内容之一。



6.3.2　ASP.NET 服务器控件概述

ASP.NET 服务器控件类似于传统的 HTML 元素，它能在内容文件（页面文件、用户控件文件或者母版页文件）中声明，也能通过编程方式实例化，并在 C#（或者其他.NET 语言）程序集中处理。

在 ASP.NET 中，Web 服务器控件“存活”在 ASPX 页面内的服务器上。当在浏览器中请求页面时，服务器端控件就由 ASP.NET 运行库（负责接收和处理 ASPX 页面请求的引擎）处理；然后控件就会写出客户端 HTML 代码，附加到最终页面输出的后面，最终出现在浏览器中用来构建页面的就是该 HTML 代码。

因此，不需要直接在页面中定义 HTML 控件，只要用下面的语法定义 ASP.NET 服务器控件即可，其中斜体部分根据控件的不同而不同：

```
<asp:TypeOfControl ID="ControlName" Runat="Server" />
```

例如，要创建一个可以显示欢迎来访者的 Label，可以用下面的语句：

```
<asp:Label ID="lblName" Runat="Server" />
```

注意： 此控件有两个属性，分别为 ID 和 Runat 属性。ID 属性用来唯一地标识页面中的一个控件，以便对它进行编程。强制性的 Runat 属性用来指出该控件存活在服务器上。如果没有这个属性，就不会处理这些控件，最终会直接显示为 HTML 源代码。对于非服务器端元素（如纯 HTML 元素），Runat 属性是可选的。如果在非服务器端控件上用了这个属性，可以通过编程代码访问它们。

可以通过页面中的内联代码或者独立的 Code Behind 文件中的代码对这个 Label 控件编程。要设置欢迎消息，可以用下面的代码（假定所用语言为 C#）：

```
lblName.Text = "欢迎你来到我的地盘！";
```

根据用途，Web 服务器控件可以分为标准控件、数据控件、导航控件、登录控件、WebParts 和 Ajax 扩展。

1．标准控件

“标准”类别中含有很多基本控件，几乎所有 Web 页面都需要它们，如上面介绍的 Label 控件。图 6-21 显示了“标准”类别中的所有控件。

由于很多控件本身会表明它们的用途，因此下面不再详细描述所有控件，仅简要强调几个重要控件。

（1）简单控件

简单控件包括 Button、Label、HyperLink、RadioButton 和 CheckBox。在工具箱中，它们的图标提供了很好的线索，可以猜出它们在浏览器中是用来干什么的。

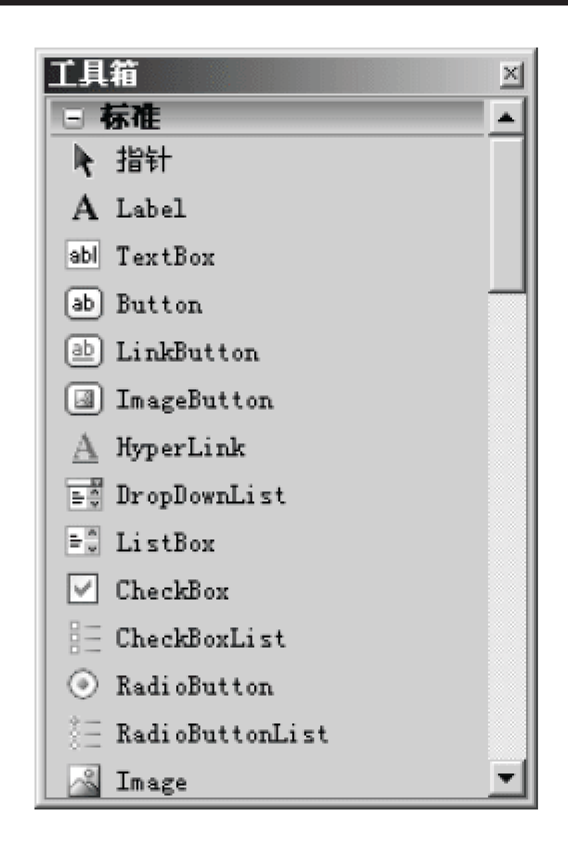


图 6-21　标准控件

（2）列表控件

工具箱中含有若干在浏览器中表现为列表的控件，包括 ListBox、DropDownList、CheckBoxList 和 RadioButtonList。要向列表中添加项目，可以在控件的起始和结束标记之间定义<asp:ListItem>元素，例如：

```
<asp:DropDownList ID="dplGroup" runat="server" Height="16px" Width="143px">
            <asp:ListItem>同学</asp:ListItem>
            <asp:ListItem>同事</asp:ListItem>
            <asp:ListItem>亲戚</asp:ListItem>
            <asp:ListItem>家人</asp:ListItem>
            <asp:ListItem>其他</asp:ListItem>
</asp:DropDownList>
```

（3）容器控件

开发 ASP.NET 应用程序时，可以把内容相关的控件（及其他标记）放在某个容器控件中，如 Panel、PlaceHolder、MultiView 和 Wizard 可以放在一起，可以用 Panel 控件同时隐藏或显示几个控件。

（4）其他标准控件

除了上述几种控件，工具箱中还有其他标准控件，如 UpLoad 控件等，详见相关文献。

2．数据控件

数据控件提供了非常方便的方式来访问各种数据源（如数据库、XML 文件与.NET 对象）并显示数据，分为数据源控件（如 SqlDataSource、ObjectDataSource、SiteMapDataSource 和 LinqDataSource 等）和数据绑定控件（如 GridView、DetailsView、FormView 和 ListView 等）。

3．导航控件

“导航”类别下的控件用来让用户找到在站点中浏览的路径。TreeView 控件和 Menu 控件用来表现数据的层次结构，并且可以用来显示站点的结构，从而可以轻松地访问站点中的所有页面。SiteMapPath 控件是一个站点导航控件，用于反映 SiteMap 对象所提供的数据。



4．登录控件

登录控件便于开发者实现用户登录验证、用户管理及相关功能，主要包括 Login、LoginName、LoginStatus、LoginView、PasswordRecovery、ChangePassword 和 CreateUserWizard 等控件。

5．Ajax 扩展

VWD 2008 中，Ajax 已经完全集成到了.NET Framework 和 VWD IDE 中，从而可以轻松地访问 Ajax 丰富的功能集，创建无闪烁的 Web 应用程序。

6．WebParts

使用 WebParts 控件可以构建高度灵活和个性化的 Web 站点。

6.3.3　使用 ASP.NET 服务器控件

1．在页面中定义服务器控件

在 VWD 2008 中，有两种途径来定义服务器控件，即在“源”视图中输入控件标记和在“设计”视图中从工具箱拖拽控件到页面指定位置。

首先新建一个个人信息管理网站，然后添加 Contact.aspx 页面，并添加 1 个 3 行 2 列的 HTML 表格，上下两行添加 div 标记及文本，中间左边摆放 1 个 TreeView 控件、1 个 SiteMapDataSourse 控件及对应的 Web.sitemap 文件；中间右边摆放 1 个 5 行 2 列的 HTML 表格，并在其中放置 3 个 TextBox 控件、1 个 DropDownList 控件和 1 个 Button 控件，如图 6-22 所示。

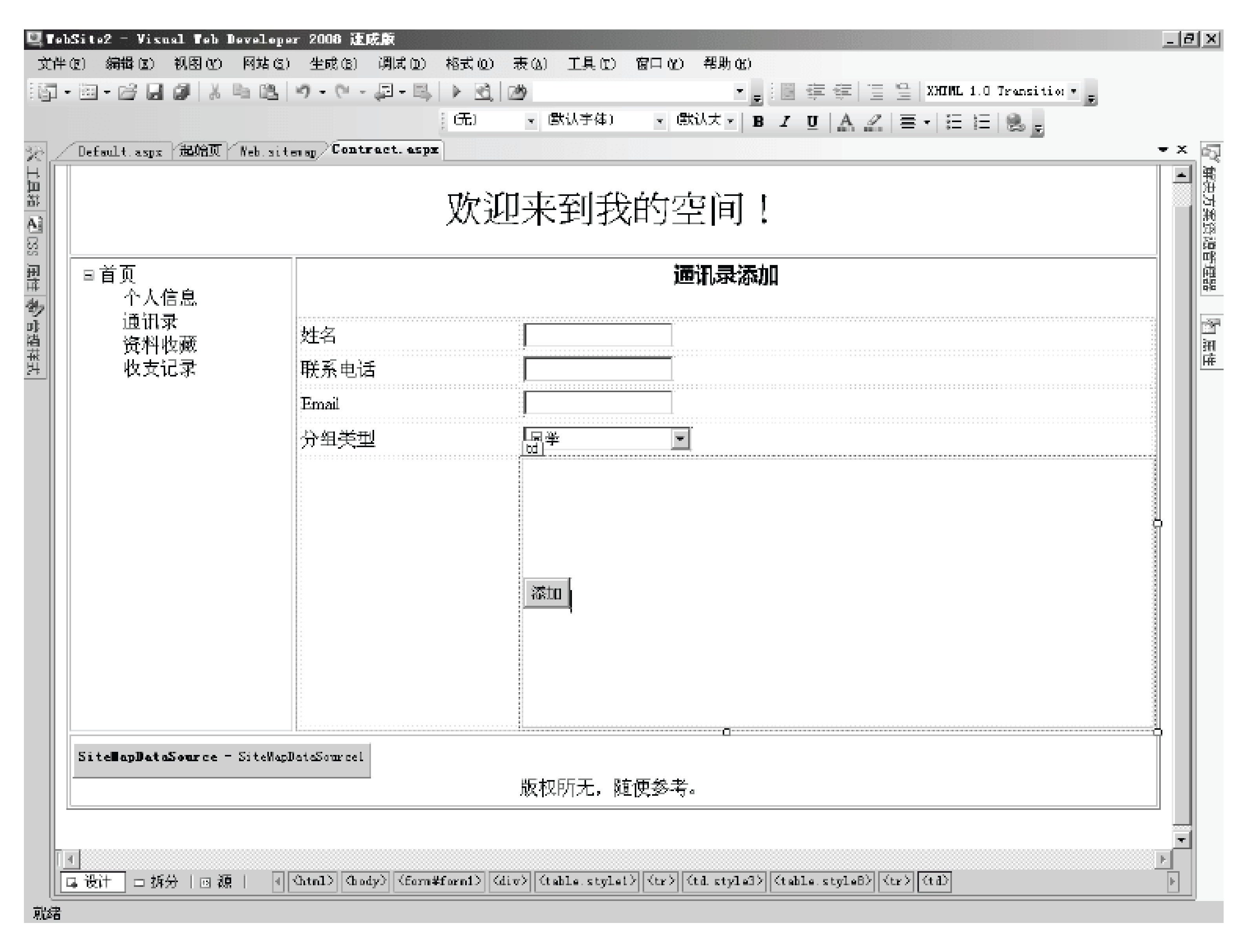


图 6-22　在页面中添加 HTML 标记和 Web 服务器控件

2. 访问控件属性

VWD 工具箱中的绝大多数服务器控件都有一些共同的行为。例如，每个控件都有一个 ID，用来在页面中唯一地标识它；还有一个 Runat 属性，总是设置为 Server，表示应在服务器上处理控件。除了这些属性外，很多服务器控件还有更多共同的属性。表 6-3 列出了最常见的属性及说明。

表 6-3 服务器控件常见属性

属　性	说　明
AccessKey	允许开发人员设置一个键，按下该键就可以在客户机中访问控件
BackColor ForeColor	允许开发人员修改浏览器中背景的颜色（BackColor）和控件文本的颜色（ForeColor）
BorderColor BorderStyle BorderWidth	修改浏览器中控件的边框，这 3 个 ASP.NET 属性中的每一个都直接映射到它的 CSS 部分
CssClass	用来定义浏览器中控件的 HTML 类属性
Enabled	确定用户是否可以与浏览器中的控件交互。例如，如果文本框是禁用的（Enabled="false"），就不能修改它的文本
Font	允许定义与字体有关的各种设置，如 Font-Size、Font-Name 和 Font-Bold
Height Width	确定浏览器中控件的高度和宽度
TabIndex	设置 HTML tabindex 属性，确定用户按 Tab 键时焦点沿着页面中控件移动的顺序
ToolTip	允许设置浏览器中控件的工具提示。这个工具提示在 HTML 中被呈现为标题属性，当用户把鼠标悬停在相关 HTML 元素上时就会显示出来
Visible	确定是否将控件发送给浏览器

图 6-22 所对应的源视图程序如下，可以看出除了许多 HTML 标记之外，服务器控件会或多或少地使用一些属性来修改其默认行为和外观，没有列出的属性按系统默认值处理。服务器控件的初始属性值有两种设置方式：通过属性面板设置（如图 6-23 所示）和在页面标记中借助智能感知（Intelligence）手工完成（如图 6-24 所示）。

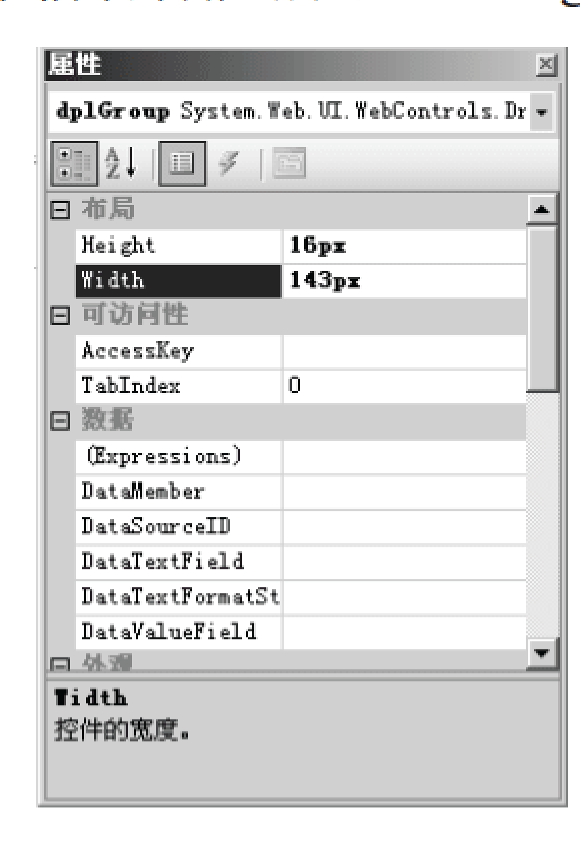


图 6-23 属性面板

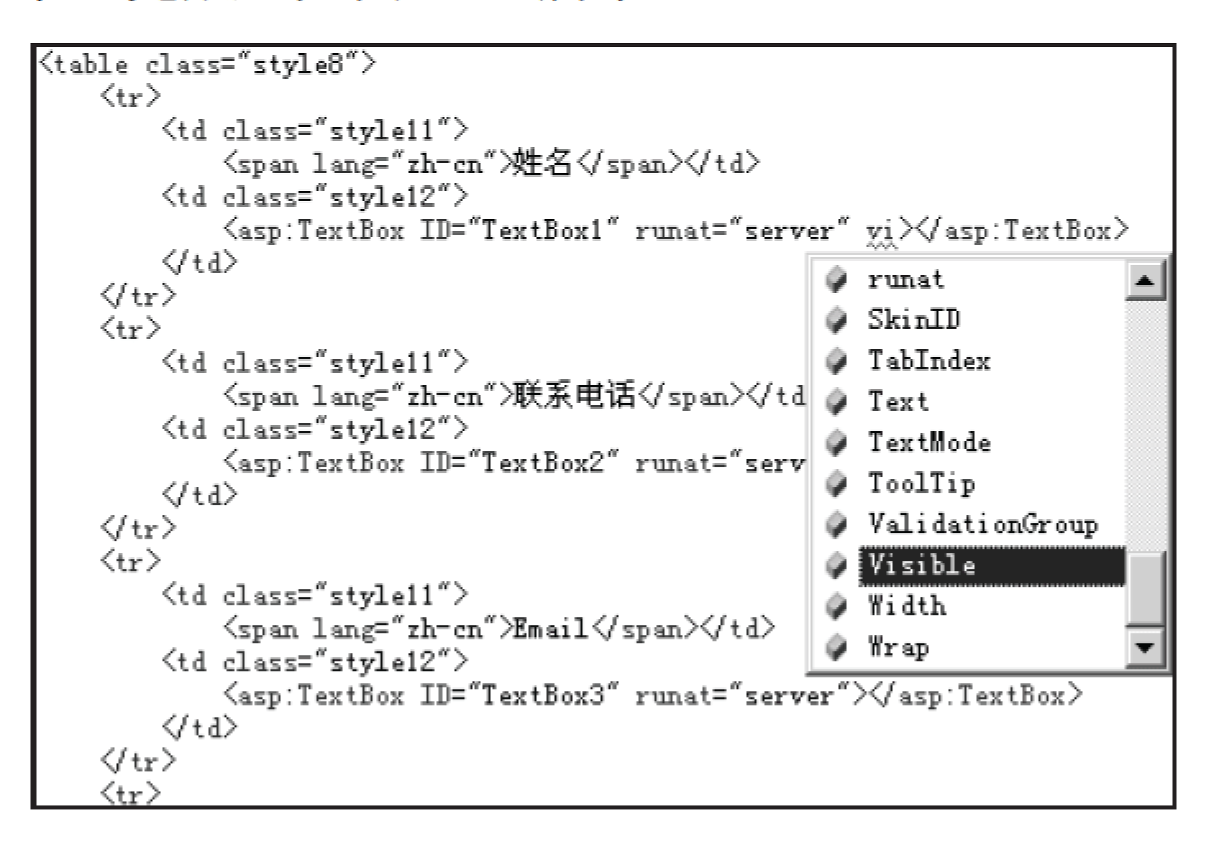


图 6-24 VWD 2008 智能感知



```
<tr>
    <td class="style11">
     <span lang="zh-cn">联系电话</span></td>
    <td class="style12">
      <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
    </td>
</tr>
<tr>
    <td class="style11">
        <span lang="zh-cn">Email</span></td>
    <td class="style12">
        <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>
     </td>
</tr>
<tr>
     <td class="style13">
         <span lang="zh-cn">分组类型</span></td>
     <td class="style14">
         <asp:DropDownList ID="dplGroup" runat="server" Height="16px" Width="143px">
           <asp:ListItem>同学</asp:ListItem>
           <asp:ListItem>同事</asp:ListItem>
           <asp:ListItem>亲戚</asp:ListItem>
           <asp:ListItem>家人</asp:ListItem>
           <asp:ListItem>其他</asp:ListItem>
         </asp:DropDownList>
    </td>
</tr>
```

3．指定 ASP.NET 控件的样式

在 ASP.NET 控件的属性中有一类特殊的属性即样式属性，所以也很容易使用 CSS 给它们指定样式，指定样式的标准方法是为控件中与样式相关的属性指定值。标准的 ASP.NET 控件默认使用内嵌 CSS 样式，例如：

```
<asp:Button ID="Button1" runat="server" BackColor="#669999"
BorderColor="Silver" BorderStyle="Solid" BorderWidth="2px"
Font-Bold="True" Font-Size="Medium" ForeColor="Red" Text="确认" />
```

ASP.NET 处理包含这个控件的 Web 页面时，会把按钮转换为标准的 HTML Input 标记，并把已设置的样式属性转换为 CSS 样式，将它们应用于 Input 标记。按钮显示的 HTML 和 CSS 如下：

```
<input type="submit" name="Button1" value="确认" id="Button1" style="color:Red;background-color:#669999;border-color:Silver;
border-width:3px;border-style:Solid;font-size:Large;font-weight:bold;" />
```

直接在 ASP.NET 控件上设置样式属性是指定样式的一种快捷方法。另外，也可以在运行期间用代码设置它们，因为这些都是控件上的标准属性。

```
this.Button1.BackColor =
        System.Drawing.ColorTranslator.FromHtml("#669999");
this.Button1.BorderColor = System.Drawing.Color.Silver;
this.Button1.BorderStyle = BorderStyle.Solid;
this.Button1.BorderWidth = Unit.Pixel(2);
this.Button1.Font.Bold = true;
this.Button1.Font.Size = FontUnit.Medium;
this.Button1.ForeColor = System.Drawing.Color.Red;
```

使用属性设置样式信息很简单、方便，但存在一些缺点，尤其是在对较多的相同控件使用相同的技术时，样式难以重用和维护，并生成大量重复 HTML 标记等。

幸好，每个 ASP.NET 服务器控件都有 CssClass 属性，可以为控件提供一个或多个类选择器规则，解决让控件显示内嵌样式带来的问题。下面的程序说明了 CssClass 属性的使用方法：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link href="StyleSheet.css" rel="stylesheet" type="text/css" />
    <style type="text/css">
        .OK
        {
            font-family: 微软雅黑;
            font-size: large;
            font-weight: bold;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>
     <asp:Button ID="Button1" runat="server" Text="Button" CssClass="OK" />
    </div>
    </form>
</body>
</html>
```

上面代码中，按钮应用了 OK 样式表类。也可以使用外部样式表中的样式，但必须附加样式表文件。



4. 服务器控件的事件处理

每一个 ASP.NET Web 页面和服务器控件都由类来实现(如 Button 控件由 Button 类实现等)，这些类自 System.Web.UI.Control 类派生。所有页面和控件都继承了 Control 类的属性、方法和事件。

在 ASP.NET 中，对象可以触发事件，而其他对象可以定义事件处理程序。例如，单击按钮触发 Click 事件，那么页面则可以用一个方法来处理按钮的单击事件（如 Button1_Click）。在 ASP.NET 应用程序中，事件通常都在客户端触发（如用户单击浏览器上的某个按钮），但在服务器上处理。用户在浏览器上对服务器控件所执行的任何行为都可能触发事件。服务器端代码响应事件，并运行存储在事件处理方法中的代码。本章采用 C#作为服务器端编程语言（读者也可以选择 VB.NET），详细语言规范可以参考相关资源。

例 6.3　在例 6.2 的基础上进行修改，让用户在文本框中输入姓名、联系电话和 E-mail，并选择分组类型，单击“添加”按钮后，在页面上将通讯录信息显示出来。

为了实现上述功能，需要添加一个 Label 控件，并且为按钮的单击事件指定一个事件处理程序，步骤如下：

（1）设置相关控件的 Name 属性值分别为 tbxName（姓名）、tbxTel（联系电话）、tbxEmail（E-mail）、dplGroup（分组类型）、lblShowInfo（结果显示）和 btnAdd（添加）。

（2）切换到页面的“设计”视图，并双击“添加”按钮控件。

（3）打开代码隐藏文件 Contact.aspx.cs，并完善 Click 事件处理代码，具体如下：

```
protected void btnAdd_Click(object sender, EventArgs e)
{
    lblShowInfo.Text = tbxName.Text + tbxTel.Text + tbxEmail.Text + dplGroup.Text;
}
```

（4）选择工作区上方的选项卡，切换到内容文件 Contact.aspx。然后单击工作区底部的“源”按钮，转到“源”视图。btnAdd 控件上添加了 OnClick 属性，该属性值已经设置成上述所添加文本行的事件处理程序的方法名 btnAdd_Click。

（5）按 F5 键或者选择 Debug→Start 命令，可运行网页。在对应文本框中输入信息，选择分组类型，然后单击“添加”按钮，页面上将会显示所输入和选择的通讯录内容。

说明：本例中只用到了少量几个属性，实际上每种服务器控件都有丰富的属性可供选择，以满足不同的要求，详细介绍请参阅相关资源。

6.4　数据库驱动的 ASP.NET 编程

在前面的内容中创建了网页，但是页面并没有与数据交互，通讯录数据并没有真正保存起来。如何从数据库获得数据并呈现在页面上，或者从页面收集数据保存到数据库中呢？本节将介绍基本的 T-SQL 编程和 ADO.NET 技术，并以 GridView 控件为例说明利用数据绑

定技术显示和更新数据的方法。

6.4.1 利用 SQL 及存储过程处理数据

在 ASP.NET 项目中可以使用多种不同类型的数据库，包括 Access、SQL Server、Oracle 和 MySQL。不过，在 ASP.NET Web 站点中最常用的数据库是微软的 SQL Server。本章所有示例都是基于 SQL Server 数据库的，读者可以采用正式版的 SQL Server 或 Microsoft SQL Server Express Edition 来调试程序。

1. 创建数据库

右击解决方案，在弹出的快捷菜单中选择“添加新项”命令，在打开的对话框中选择“SQL Server 数据库”选项，如图 6-25 所示，并指定名称为 PersonMIS.mdf，单击“添加”按钮添加数据库。

说明：必须安装 SQL Server Express 组件后才可以使用该功能，否则必须在正式版的 SQL Server 中创建数据库。

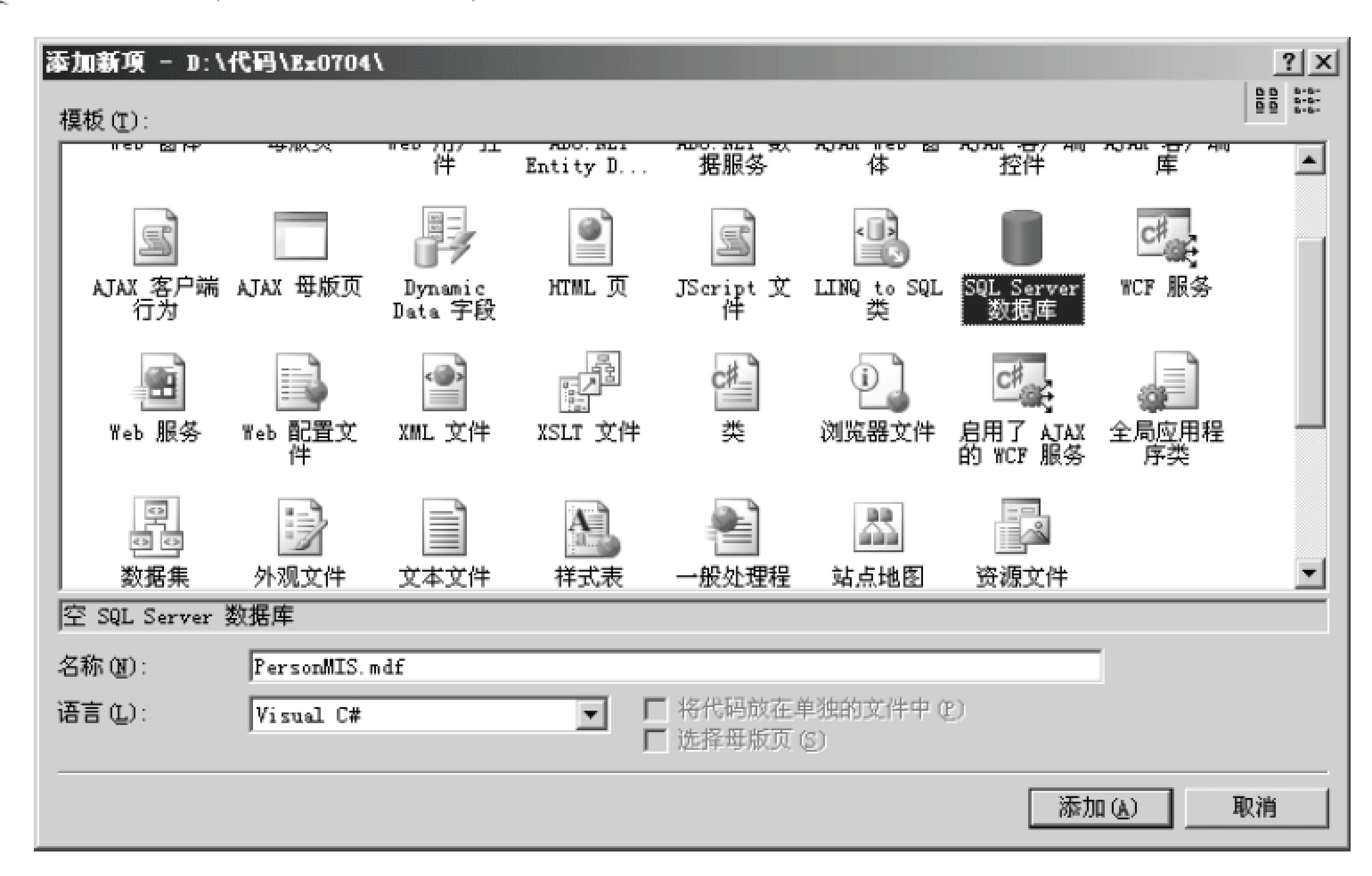


图 6-25 添加 SQL Server 数据库

2. 添加数据表

为了存储 6.3.3 节中介绍的通讯录内容，需要按表 6-4 中的说明创建数据表。

表 6-4 通讯录（Contacts）表结构说明

字段名称	数据类型	长度	是否允许为空	字段描述	备注
SN	INT		否	序号	主键，标识字段
Name	VARCHAR	20	否	姓名	
Phone	VARCHAR	40	是	联系电话	
Email	VARCHAR	40	是	E-mail	
GroupName	VARCHAR	10	否	所属分组	



3．使用 T-SQL 语言检索和操纵数据

与数据库交互时，需要花大量时间检索和操纵数据，这些操作可归结为 4 种不同类型，即 CRUD——Create（创建）、Read（读取）、Update（更新）和 Delete（删除）。这些数据操作很重要，也是数据库相关课程的核心内容，下面简要介绍这些操作的 SQL 语句的特殊形式。

（1）查询数据

查询语句一般包括选择（SELECT）子句、条件（WHERE）子句、排序（ORDER BY）子句和分组（GROUP BY）子句等几部分。其中选择子句返回必需的字段列表，条件子句用来筛选数据，排序子句用于将查询记录排序，分组子句用于数据汇总，这些子句的构造方式要根据实际系统需求而定。这里介绍两个特殊并且用途广泛的查询语句，一是查询全部记录，它不需要选择条件，这里假定返回所有字段并且不考虑排序和分组情况，代码如下：

```
SELECT SN, Name, Phone, Email, GroupName
FROM Contacts
```

二是查询单个记录，它与查询全部记录类似，只是附加了一个“按主键查询”的条件子句，代码如下：

```
SELECT SN, Name, Phone, Email, GroupName
FROM Contacts
WHERE SN = 1
```

（2）添加单个记录

要将新数据插入到 SQL Server 表中，可以使用 INSERT 语句，它有一些不同的形式，但最简单的形式如下所示：

```
INSERT INTO TableName (Column1 [, Column2]) VALUES (Value1 [, Value2])
```

下列代码段显示了如何将新值插入到 Genre 表中：

```
INSERT INTO Contacts (Name, Phone, Email, GroupName)
VALUES ('章章', '13512341234', 'zhangzhang@163.com', '同学')
```

在上面语句中，并没有指定 SN 字段的值，因为该字段是标识字段，在插入数据时自动生成，假定为 1。

（3）更新单个记录

要更新表中的数据，可以使用 UPDATE 语句，例如：

```
UPDATE TableName
SET Column1 = NewValue1 [, Column2 = NewValue2]
WHERE Column3 = Value3
```

为了确保只更新特定数据，更新记录时一般需要指定更新条件，更新单个记录时往往

按主键或唯一键来构造条件子句。下列代码段显示了如何更新指定的记录：

```
UPDATE Contacts
SET Name = '章章', Phone='10086',    Email = 'zhangzhang@163.com', GroupName = '同学'
WHERE SN = 1
```

（4）删除单个记录

删除数据库记录使用 DELETE 语句，和 SELECT 和 UPDATE 语句一样，可以在 DELETE 语句中使用 WHERE 子句来限制删除的记录数。这个 WHERE 子句通常非常重要，因为如果没有它，很可能会删除整个表而不只是一小部分记录。

下列代码段显示了如何按主键删除指定的记录：

```
DELETE Contacts
WHERE SN = 1
```

4．使用存储过程完成数据操作

许多实际的应用程序使用存储过程来与 SQL Server 或者其他大型数据库交互，调用存储过程和直接调用 SQL 语句具有相同效果。调用存储过程的优点在于，在大型应用程序中，存储过程可更加高效，因为数据库对存储过程的执行进行了优化。

上述 4 类简单的 SQL 语句都可以被封装为相应的存储过程，并根据需要指定存储过程所需参数。

（1）查询全部记录

```
CREATE PROCEDURE ContactsSelectAll
AS
        SELECT SN, Name, Phone, Email, GroupName
        FROM Contacts
```

（2）查询单个记录

```
CREATE PROCEDURE ContactsSelectOne
        @SN INT
AS
        SELECT SN, Name, Phone, Email, GroupName
        FROM Contacts
        WHERE SN = @SN
```

（3）添加单个记录

```
CREATE PROCEDURE ContactsInsertOne
        @Name          VARCHAR(20),
        @Phone         VARCHAR(40),
        @Email         VARCHAR(40),
        @GroupName  VARCHAR(10)
AS
```



```
        INSERT INTO Contacts (Name, Phone, Email, GroupName)
        VALUES (@Name, @Phone, @Email, @GroupName)
```

（4）更新单个记录

```
CREATE PROCEDURE ContactsUpdateOne
        @SN              INT,
        @Name            VARCHAR(20),
        @Phone           VARCHAR(40),
        @Email           VARCHAR(40),
        @GroupName  VARCHAR(10)
AS
        UPDATE Contacts
        SET Name = @Name,
              Phone = @Phone,
              Email = @Email,
              GroupName = @GroupName
        WHERE SN = @SN
```

（5）删除单个记录

```
CREATE PROCEDURE ContactsDeleteOne
        @SN INT
AS
        DELETE Contacts
        WHERE SN = @SN
```

6.4.2　ADO.NET 技术概述

ADO.NET 是 Microsoft .NET 框架中所包含的一组库，用于在.NET 应用程序中同各种数据存储区进行通信。在.NET 框架中，ADO.NET 类库位于 System.Data 命名空间下，这些类库包括连接到数据源、执行命令以及存储、操作和获取数据等功能。还可将 ADO.NET 用作一种可靠、分层的非连接数据缓存，以脱机方式处理数据。利用最主要的非连接对象 DataSet，可以对数据进行排序、检索、筛选、保存以及在分层数据中进行浏览。

1. ADO.NET 对象模型

ADO.NET 对象模型是很丰富的，然而它的核心仅包括一些比较简单的类集，可分为两类：一类为“连接的”对象，另一类为“断开连接的”对象，后者允许将查询结果保存在内存中进行处理。

DbConnection 对象代表指向数据源的连接通道，可以在不同的 DbCommand 对象间共享，而且支持事务。

DbCommand 对象会把命令（通常是 SQL 语句或存储过程名）发送给数据库，以完成

应用程序的特定操作。

DataSet 类用于表示一个数据库的富子集，它缓存在会话状态和内存中，且支持离线操作，不需要不间断的数据库，只需要周期性地让 DataSet 重新连接数据库，这样才可以把 DataSet 的更新提交给数据库，同时，也可以把由其他进程对数据库的更新放入 DataSet 中。

DataSet 被认为是“在内存中的数据库”，因而可以存放一系列表及其数据，它使用必要的元数据来表示数据库表之间的关系和约束。DataSet 由 DataTable 对象和 DataRelation 对象组成，而 DataTable 对象又包括 DataRow 对象和 DataColumn 对象。

ADO.NET 使用 DataAdapter 对象在 DataSet 对象和数据库之间实现交互，避免了 DataSet 对象与数据库架构间过于紧密的联系，进而保证了一个 DataSet 可表示多于一个数据库或其他数据源。

ASP.NET 提供了不同版本的 DataAdapter 对象，分别与不同类型数据源（如 SQL Server、Access、Oracle 等）“打交道”，这些版本实际上就是 DataAdapter 类的子类对象。

DataReader 对象是 DataSet 对象的一个轻量级替代品，用于在单向只读访问数据时提供更好的系统性能，它可以执行 SQL 语句或者存储过程。

2. .NET 数据提供程序

.NET 数据提供程序是一个类的集合，专门用于同特定类型的数据存储区进行通信。.NET 框架包括以下 4 种提供程序：

- SQL Client .NET 数据提供程序，专门用于访问 SQL Server 数据库。
- Oracle Client .NET 数据提供程序，专门用于访问 Oracle 数据库。
- ODBC .NET 数据提供程序。
- OLE DB .NET 数据提供程序。

ODBC 和 OLE DB .NET 数据提供程序经常被称为“桥梁”组件，以便使开发人员能够分别通过 ODBC 驱动程序和 OLE DB 提供程序同各种数据存储区进行通信。

每个.NET 数据提供程序都有自己的命名空间和一套相关类，是 System.Data 命名空间的一个子集。其中，SQL Client 数据提供程序位于 System.Data.SqlClient 命名空间中，ODBC .NET 数据提供程序位于 System.Data.Odbc 命名空间中；OLE DB .NET 数据提供程序位于 System.Data.OleDb 命名空间中；Oracle Client .NET 数据提供程序则位于 System.Data.OracleClient 命名空间中。

由于每个.NET 数据提供程序都实现相同的基本接口，所以在本书中不需要针对所有.NET 数据提供程序来解释如何使用这些接口，而是着重介绍 SQL Client .NET 数据提供程序及其包括的 SqlConnection、SqlCommand、SqlAdapter、SqlDataReader 对象。

6.4.3 使用 ADO.NET 技术访问 SQL Server 数据库

在开发 ASP.NET 应用程序时，访问数据库相当方便，既可以通过可视化配置方式轻松实现数据库操作，也能够以编程方式实现相应操作，从而提高程序架构的清晰度和可维护



性。本小节主要讨论以编程方式访问数据库的基本内容。

1．关键步骤

使用 ADO.NET 技术进行数据查询和数据操纵一般包括连接数据库、配置命令、执行命令并处理执行结果几个步骤。

（1）连接数据库

有两种方式可以在运行时生成 SqlConnection 对象：既可以使用无参数构造函数简单地生成一个未初始化的 SqlConnection 对象，也可以使用连接字符串作为参数来构造，其中连接字符串有 3 种形式，具体取决于数据库验证方式和数据库实例类型，例如：

```
SqlConnection conn1 = new SqlConnection();//构造默认的连接
SqlConnection conn2 = new SqlConnection("Data Source=.; Initial Catalog=PersonMIS; Integrated
Security=True;");//Window 集成身份验证，使用独立版本数据库
SqlConnection conn3 = new SqlConnection("Data Source=.; Initial Catalog=PersonMIS; User ID=sa;
Password=abc; connect Timeout=30");//混合身份验证，使用独立版本数据库
SqlConnection conn4 = new SqlConnection("Data Source=.\SQLEXPRESS; AttachDbFilename=
|Data Directory|\PersonMIS.mdf; Integrated Security=True; User Instance=True ");//使用 Express 版本数
据库
```

在生成 SqlConnection 对象时，其初始状态为“关闭”（Closed），并没有真正连接到数据存储区。要连接至数据存储区，首先利用构造函数或者通过设置该对象的 ConnectionString 属性来提供一个有效的连接字符串，然后调用该对象的 Open 方法。如果在 SqlConnection 关闭状态下尝试执行查询命令，系统将会发生 InvalidOperationException 异常。

为了合理利用数据库连接资源，在使用完 SqlConnection 对象时，要确保关闭它，只需要调用该对象的 Close 方法即可。

在已经连接到 SQL Server 数据库之后，SqlConnection 对象可以用作许多操作的起始点——创建命令、开始事务以及提取架构信息。

（2）配置命令

SqlCommand 类是用于对 SQL Server 数据库进行访问的命令类。创建 SqlCommand 对象有 3 种方式：一是使用 new 关键字直接创建对象的一个新实例，然后设置适当属性；二是使用一个可用的构造函数来指定查询字符串以及 SqlConnection 对象；三是调用 SqlConnection 类的 CreateCommand 方法来简化创建过程，这 3 种方法对应的程序如下：

```
string strSQL;
strSQL = "SELECT CustomerID, CompanyName FROM Customers";
conn4.Open();
SqlCommand cmd;
//使用无参数构造函数
cmd = new SqlCommand();
cmd.Connection = conn4;
cmd.CommandText = strSQL;
//使用参数化构造函数
cmd = new SqlCommand(strSQL, conn4);
```

```
//使用 Connection 对象的 CreateCommand 方法
cmd = conn4.CreateCommand();
cmd.CommandText = strSQL;
cmd.CommandType = CommandType.Text;//配置命令类型
```

利用 SqlCommand 对象，调用存储过程就变得简单。将 CommandText 属性设置为存储过程的名称，将 CommandType 属性设置为 CommandType.StoredProcedure，代码如下：

```
cmd.CommandType = CommandType.StoredProcedure;
cmd.CommandText = "ContactsSelectAll";
```

参数化查询（Parameterized Query 或 Parameterized Statement）是指在访问数据时，在需要给命令对象附加数值或数据的地方，使用参数传值的方式代替，可以有效预防 SQL 注入攻击（SQL Injection）。

在实际应用中，希望使用参数化查询的情景有很多，如根据用户输入或选择的条件进行检索和删除数据、在添加数据时向数据库传递新数据及在修改数据时向数据库传递新值和修改条件等。根据传值的方向，参数可分为输入参数、输出参数和返回值参数，由 SqlParameters 对象的 Direction 属性来指定。

在.NET 2.0 以后，添加参数的方式非常简单，只需调用 AddWithValue 方法即可，代码如下：

```
cmd.Parameters.AddWithValue("@SqlParamName", DotNetParamName);
```

（3）执行命令和处理返回值

配置完成后，只需要利用 SqlCommand 的适当方法来调用存储过程即可。如果希望查看返回的行，则使用 ExecuteReader；如果希望仅获取单一值，则使用 ExecuteScalar；如果希望获取被修改的行数或者对于返回数据不感兴趣，则使用 ExecuteNonQuery。

- 使用 SqlDataReader 对象读取数据

调用 SqlCommand 对象的 ExecuteReader 方法会返回 SqlDataReader 对象，可以用它来获取由这些查询返回的行。在调用 ExecuteReader 之后，第 1 行数据并非立即可用，所以在读取结果的第 1 行之前必须调用 Read 方法。对 Read 方法的后续调用会依次获取下一行，直到读完所有数据。此方法还返回一个 Boolean 值，以指示 SqlDataReader 是否有可用行。因此，当 Read 方法返回 False 时，表明已经到达结果的末尾，此时，应该关闭 SqlDataReader 对象。

通过 SqlDataReader 对象读取数据时，往往需要将数据行转换为实体类对象，以便进一步处理（详见后续内容）。

- 使用 SqlDataAdapter 对象读取数据

SqlDataAdapter 类用作 ADO.NET 对象模型中已连接部分和未连接部分之间的桥梁，可以使用 SqlDataAdapter 对象从数据库中获取数据，并将其存储在 DataSet 中，也可以借助 SqlDataAdapter 对象提取 DataSet 对象的数据变化，并更新到数据库。

在创建 SqlDataAdapter 时，通常需要将 SelectCommand 属性设置为一个有效的 SqlCommand 对象，调用 SqlDataAdapter 类的 Fill 方法会执行该查询，并将结果填充到 DataSet 对象中。



（4）执行操作查询

不返回结果集的查询通常被称为操作查询（action query），分为以下两大类。

- 数据操作语言（DML）查询

这些查询修改数据库的内容，也称为基于查询的更新（QBU），包括针对数据表的 INSERT、UPDATE 和 DELETE 操作。

- 数据定义语言（DDL）查询

这些查询修改数据库的结构，包括针对数据库对象的 CREATE 和 DROP 操作。

SqlCommand 允许将受此查询影响的行数作为 ExecuteNonQuery 方法的返回值返回，以此确定操作是否成功。

说明： *ExecuteNonQuery 方法的返回值不是判断操作成败的唯一标准，甚至在 SQL Server 启用 NOCOUNT 选项后，返回值将会失效。一个推荐的处理方案是将操作成败的判断放在存储过程中，通过 RAISERROR 函数与前台应用程序的异常处理结合起来，构成一个错误处理体系。*

2．通讯录实体类代码

在更新和插入数据时，需要给存储过程传递的参数会很多，基本上是表的全部字段，这时如果将每一个参数值都声明为一个形参，会大大增加成员方法接口的复杂性。在分层架构应用程序中，一个推荐的替代方案就是使用实体对象作为参数，这也符合面向对象方法的基本思想。

实体对象的另一个用途是作为查询单个记录方法的返回值。设计实体类时，必须考虑类成员与表字段的对应关系和成员数据类型，如 Name 字段在 SQL Server 中为 VARCHAR 类型，在.NET 中对应类型为 String。

在 VWD 2008 中添加实体类的步骤如下：

（1）右击网站 Ex0704，在弹出的快捷菜单中选择“添加新项”命令，弹出如图 6-26 所示的对话框。

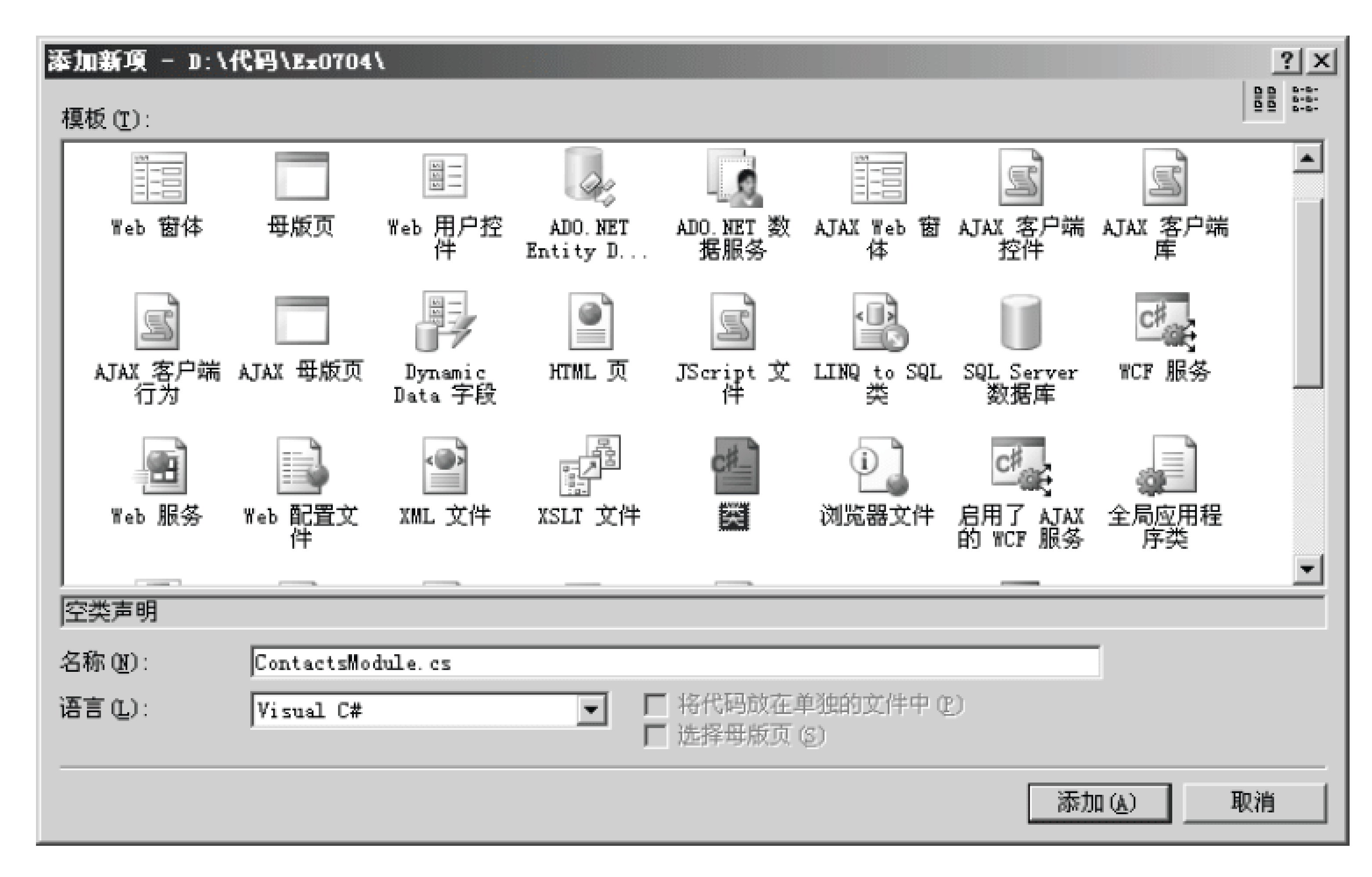


图 6-26　添加新项

（2）选中“类”模块，输入类名称“ContactsModule.cs”，然后单击“添加”按钮。

（3）ASP.NET 中类文件应该放在 App_Code 系统文件夹内，如果是第一次添加类文件，会弹出如图 6-27 所示的对话框，单击“是”按钮即可。



图 6-27　添加 App_Code 文件夹

（4）在打开的 ContactsModule.cs 中，完善 ContactsModule 类的代码，具体代码如下所示：

```
public class ContactsModule
{
    public ContactsModule(){ }

    public     Int32 SN;//序号
    public     String     Name;//姓名
    public     String     Phone;//联系电话
    public     String     Email;//E-mail
    public     String     GroupName;//所属分组
}
```

3．通讯录数据访问类代码

虽然数据库访问代码可以与界面处理代码放在一起，但是，通常情况下这不是一个好主意，因为它很大程度上降低了代码的可读性和软件架构的灵活性。一个推荐的方案是：将数据访问代码单独存放，形成一个 ContactsDA 类，把调用存储过程的相关操作封装成相应的类成员方法。数据访问类文件 ContactsDA.cs 的添加步骤与上述方法相同，而各个成员方法的代码如下所示。

（1）查询所有记录

```
public DataTable ContactsSelectAll()
{
    using (SqlConnection conn = new SqlConnection(ConnStr))
    {
        if(ConnectionState.Closed == conn.State)
        {
            conn.Open();
        }
        using (SqlCommand cmd = conn.CreateCommand())
        {
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.CommandText = "ContactsSelectAll";
```



```
                try
                {
                    SqlDataAdapter Adapter = new SqlDataAdapter(cmd);
                    DataSet ds = new DataSet();
                    Adapter.Fill(ds);
                    conn.Close();

                    return ds.Tables[0];
                }
                catch (SqlException sqlEx)
                {
                    throw sqlEx;
                }
            }
        }
    }
```

（2）查询单个记录

```
public ContactsModule ContactsSelectOne(Int32 SN)
{
    using (SqlConnection conn = new SqlConnection(ConnStr))
    {
        if(ConnectionState.Closed == conn.State)
        {
            conn.Open();
        }
        using (SqlCommand cmd = conn.CreateCommand())
        {
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.CommandText = "ContactsSelectOne";

            #region 添加存储过程参数
            cmd.Parameters.AddWithValue("@SN", SN);//序号
            #endregion

            try
            {
                SqlDataReader dr = cmd.ExecuteReader();
                if (dr.HasRows == false)
                {
                    dr.Close();
                    return null;
                }

                ContactsModule ContactsMod = new ContactsModule();

                dr.Read();
```

```
                #region 读取字段值
                if (!dr.IsDBNull(0))
                {
                    ContactsMod.SN = dr.GetInt32(0);//序号
                }
                else
                {
                    ContactsMod.SN = 0;//序号
                }
                //读取其他字段
                #endregion

                dr.Close();
                conn.Close();

                return ContactsMod;
            }
            catch(SqlException sqlEx)
            {
                throw sqlEx;
            }
        }
    }
}
```

（3）删除一条记录

```
public void ContactsDeleteOne(Int32 SN)
{
    using (SqlConnection conn = new SqlConnection(ConnStr))
    {
        if(ConnectionState.Closed == conn.State)
        {
            conn.Open();
        }
        using (SqlCommand cmd = conn.CreateCommand())
        {
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.CommandText = "ContactsDeleteOne";

            #region 添加存储过程参数
            cmd.Parameters.AddWithValue("@SN", SN);//序号
            #endregion

            try
            {
                cmd.ExecuteNonQuery();
                conn.Close();
```



```
                }
                catch (SqlException sqlEx)
                {
                    throw sqlEx;
                }
            }
        }
    }
```

（4）更新一条记录

根据是否采用实体对象作为参数，ContactsUpdateOne 方法有两个版本，分别如下所示：

```
//版本 1
public void ContactsUpdateOne(ContactsModule   ContactsMod)
{
    using (SqlConnection conn = new SqlConnection(ConnStr))
    {
        if(ConnectionState.Closed == conn.State)
        {
            conn.Open();
        }
        using (SqlCommand cmd = conn.CreateCommand())
        {
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.CommandText = "ContactsUpdateOne";

                #region 添加存储过程参数
            cmd.Parameters.AddWithValue("@SN", ContactsMod.SN);//序号
            cmd.Parameters.AddWithValue("@Name", ContactsMod.Name);//姓名
            cmd.Parameters.AddWithValue("@Phone", ContactsMod.Phone);//联系电话
            cmd.Parameters.AddWithValue("@Email", ContactsMod.Email);//Email

            cmd.Parameters.AddWithValue("@GroupName",  ContactsMod.GroupName);// 所属
分组
            #endregion

            try
            {
                cmd.ExecuteNonQuery();
                conn.Close();
            }
            catch (SqlException sqlEx)
            {
                //异常处理代码
            }
        }
```

```
        }
    }
    //版本 2
    public void ContactsUpdateOne(Int32 SN, String Name, String Phone, String Email, String
GroupName)
    {
        using (SqlConnection conn = new SqlConnection(ConnStr))
        {
            if (ConnectionState.Closed == conn.State)
            {
                conn.Open();
            }
            using (SqlCommand cmd = conn.CreateCommand())
            {
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.CommandText = "ContactsUpdateOne";

                #region 添加存储过程参数
                cmd.Parameters.AddWithValue("@SN", SN);//序号
                cmd.Parameters.AddWithValue("@Name", Name);//姓名
                cmd.Parameters.AddWithValue("@Phone", Phone);//联系电话
                cmd.Parameters.AddWithValue("@Email", Email);//Email

                cmd.Parameters.AddWithValue("@GroupName", GroupName);//所属分组
                #endregion
          //其余代码与版本 1 相同
    }
```

（5）插入一条记录

与更新操作类似，ContactsInsertOne 也有两种实现方法，代码如下所示：

```
    //版本 1
    public void ContactsInsertOne(ContactsModule   ContactsMod)
    {
       using (SqlConnection conn = new SqlConnection(ConnStr))
       {
          if(ConnectionState.Closed == conn.State)
          {
              conn.Open();
          }
          using (SqlCommand cmd = conn.CreateCommand())
          {
              cmd.CommandType = CommandType.StoredProcedure;
              cmd.CommandText = "ContactsInsertOne";

              //其余与 ContactsUpdateOne 方法的版本 1 类似
          }
```



```
        }
    }
    //版本 2
    public void ContactsInsertOne (String Name, String Phone, String Email, String GroupName)
    {
        using (SqlConnection conn = new SqlConnection(ConnStr))
        {
            if (ConnectionState.Closed == conn.State)
            {
                conn.Open();
            }
            using (SqlCommand cmd = conn.CreateCommand())
            {
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.CommandText = " ContactsInsertOne ";

                #region 添加存储过程参数
                cmd.Parameters.AddWithValue("@Name", Name);//姓名
                cmd.Parameters.AddWithValue("@Phone", Phone);//联系电话
                cmd.Parameters.AddWithValue("@Email", Email);//Email
                cmd.Parameters.AddWithValue("@GroupName", GroupName);//所属分组
                #endregion
         //其余代码与 ContactsInsertOne 方法的版本 1 相同
    }
```

6.4.4　显示和更新数据

ASP.NET 内置了丰富的数据访问控件，可以有效地处理系统中的数据，根据其功能的不同分为数据绑定控件（Data-Bound Control）和数据源控件（Data Source Control），如图 6-28 所示。一般情况下，往往将两类控件结合起来，数据源控件负责提取和处理数据，主要完成连接数据源、执行数据的 CRUD 操作等功能，数据绑定控件负责把数据以多种方式显示在界面上。

1. 数据绑定控件简介

在构建软件（特别是 Web 站点）时，遇到的最普遍的问题就是将一个集合作为用户界面（UI）元素展现出来。ASP.NET 包含了很多数据绑定控件，都能够承载这些集合并显示到正确的标签中。

每一个 ASP.NET 中的数据绑定控件都包含一个属性来将其关联到一个数据源上去，这些控件都包含一个 DataSource 属性，可以将任何实现了 IEnumerable 接口的集合以及 DataSet 和 DataTable 对象指定给它。在将集合关联到控件以后，可以在页面(或控件)上调用 DataBind 方法来指示控件去迭代整个集合。

ASP.NET 包含了很多至少支持简单数据绑定的控件，包括 CheckBoxList、RadioButtonList、

DropDownList、ListBox、TreeView 和 Menu 等控件，以及 GridView、DataList、DetailView、FormView 和 Repeater 等更复杂的控件，这些控件用起来很简单，只需在代码中给它们关联一个数据源，就可以自动进行呈现，这种绑定方式称为代码式数据绑定。并且复杂控件还提供了多种样式，以便于以更精细的方式来显示数据。

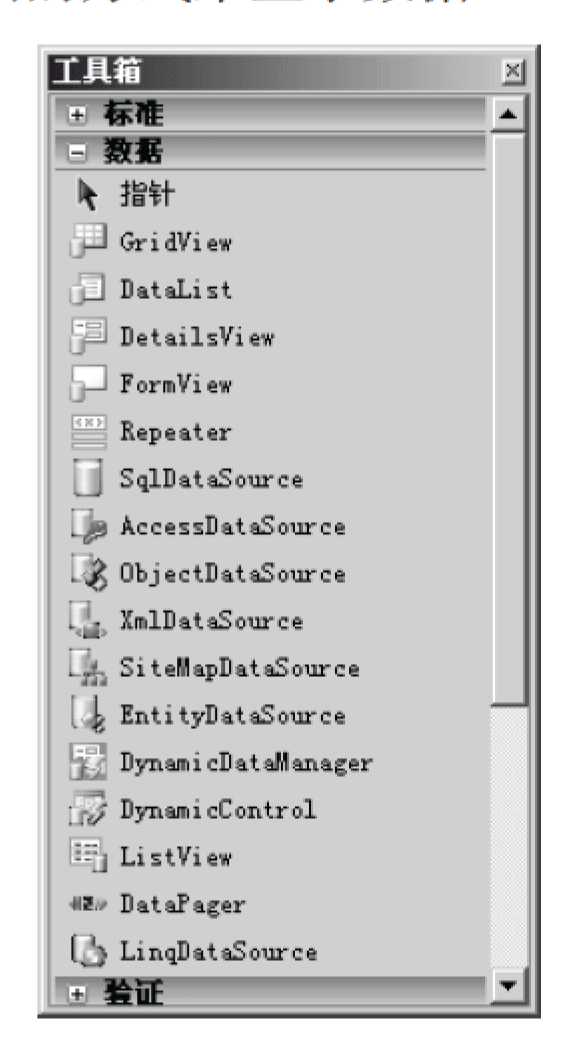


图 6-28　数据控件

2．数据源控件简介

有些数据绑定控件还包含一个 DataSourceID 属性，以支持更复杂的数据绑定，它使用一个独立的数据源控件来为数据绑定控件管理数据。数据源控件是 ASP.NET 2.0 开始引入的一个新的抽象层，这些控件抽象了底层数据提供程序的使用，例如，SQL 数据提供程序或 OLE DB 数据提供程序。此外，也可以像其他 Web 服务器控件那样使用数据源控件，例如，可以在 HTML 中明确定义和控制数据源控件的操作，或编程定义和控制它们。这种无须编写任何代码，就可以进行所有的数据访问和处理的数据绑定方法被称为声明式数据绑定。声明式绑定大大简化了数据绑定控件显示数据集合的过程，它们通过引用一个页面上的 DataSource 控件的 ID 来完成任务。

ASP.NET 3.5 中 6 个内置的数据源控件分别用于特定类型的数据访问。表 6-5 为 ASP.NET 中的每个数据源控件及其说明。

表 6-5　数据源控件及其说明

控　件　名	说　　明
SqlDataSource	允许访问支持 ADO.NET 数据提供程序的所有数据源，该控件默认可以访问 ODBC、OLE DB、SQL Server、Oracle 和 SQL Server CE 提供程序
LinqDataSource	可以使用 LINQ 查询访问不同类型的数据对象
ObjectDataSource	可以对业务对象或其他返回数据的类执行特定的数据访问
XmlDataSource	可以对 XML 文档执行特定的数据访问，包括物理访问和内存访问
SiteMapDataSource	可以对站点地图提供程序所存储的 Web 站点进行特定的站点地图数据访问
AccessDataSource	可以对 Access 数据库执行特定的数据访问



3．通讯录数据管理模块的实现

有关数据访问控件的使用方法需要至少一本书的内容才有可能介绍清楚，本节仅以通讯录管理为例介绍实际应用中联合使用 GridView 控件和 ObjectDataSource 控件的一般方法。

1）显示通讯录数据

在网页上显示所有通讯录数据的步骤如下：

（1）打开 Contract.aspx 页面，并使它处于设计状态，并从工具箱中拖拽一个 GridView 控件到 Contract.aspx 页面中，如图 6-29 所示。

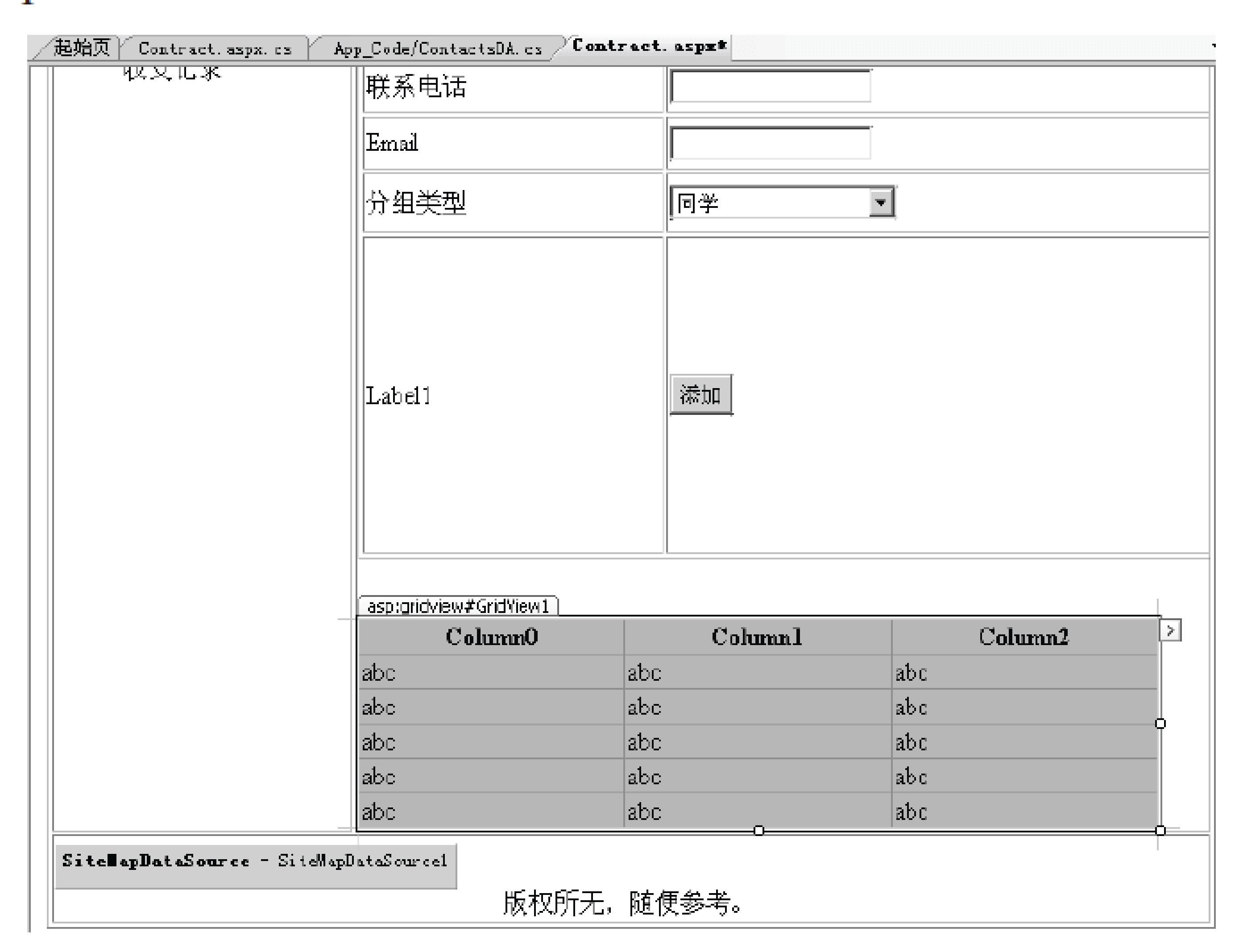


图 6-29　添加 GridView 控件

（2）以相同的方式，向页面中添加 ObjectDataSource 控件；单击该控件右上角的箭头符号，在弹出的“ObjectDataSource 任务”对话框中选择“配置数据源”选项。

（3）在配置数据源向导中，选择业务对象为 ContactsDA，如图 6-30 所示，并单击“下一步”按钮。



图 6-30　选择业务对象



（4）为数据源的 SELECT 操作指定相应的数据访问方法，如 ContactsSelectAll 方法，如图 6-31 所示，然后单击“完成”按钮结束数据源控件的配置。

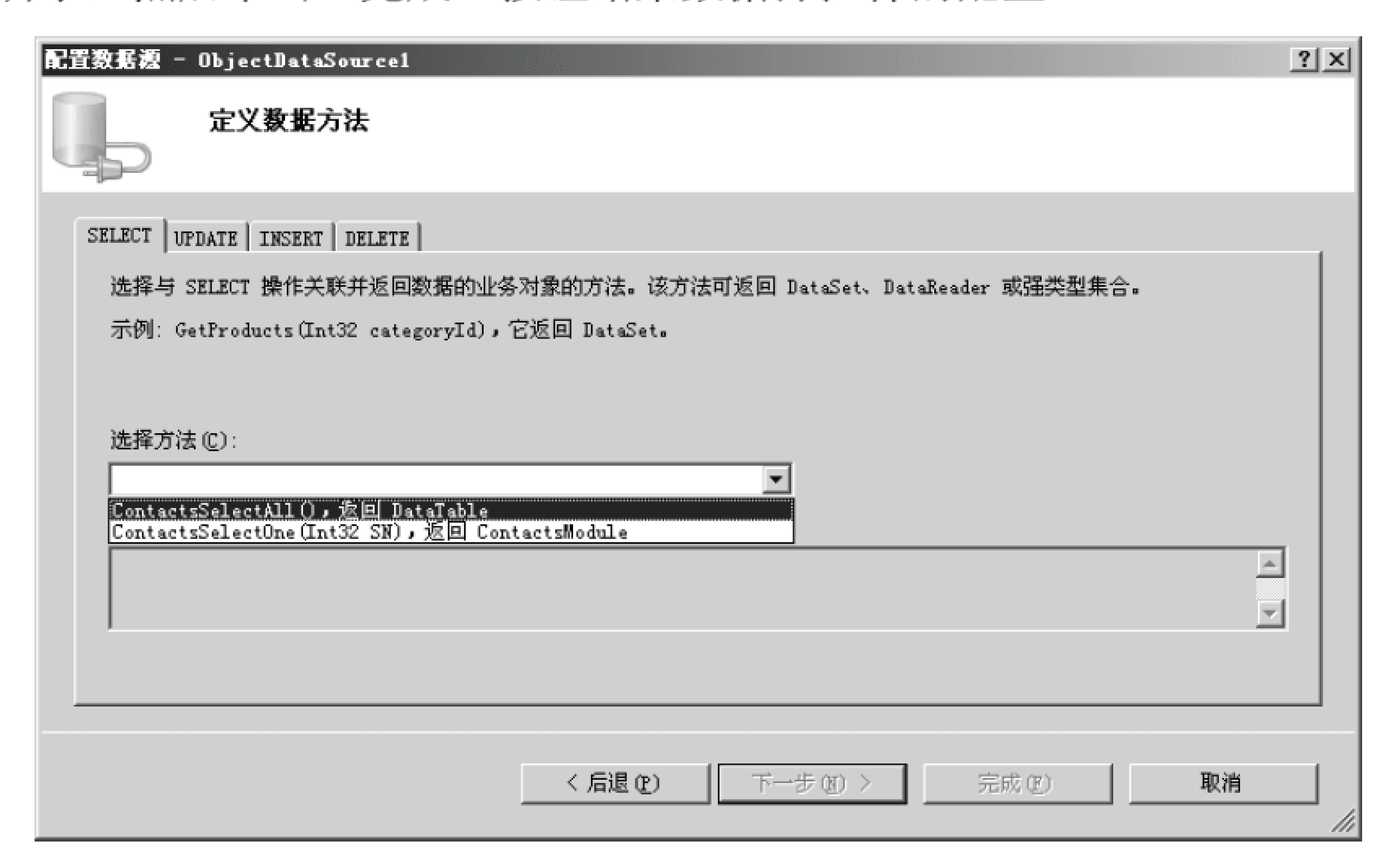


图 6-31　定义数据方法

（5）单击 GridView1 控件右上角的箭头符号，在弹出的“GridView 任务”对话框中选择 ObjectDataSource1 作为数据源。

（6）启动调试或直接运行该网站，网页上会显示数据库中的通讯录记录（如果有的话）。

2）编辑通讯录数据

如果要在网页上显示所有通讯录数据，需要在前面内容的基础上完成以下工作：

（1）按上面步骤（4）的方式配置 ObjectDataSource1 控件的 UPDATE 和 DELETE 两个命令相应的方法，分别为 ContactsUpdateOne 方法（版本 2）和 ContactsDeleteOne 方法。而 INSERT 命令可以暂时不必配置。

（2）按上面步骤（5）的方式配置 GridView1 控件，在“GridView 任务”对话框中选中“启用编辑”和“启用删除”两个复选框。同时，该控件的列表会发生一些变化，出现“编辑”和“删除”两个操作按钮。

（3）在属性面板中，将 GridView1 控件的 DataKeyNames 属性设置为 SN。

（4）启动调试或直接运行该网站，即可在网页上修改和删除通讯录记录（如果有的话）。

3）以编程方式添加通讯录数据

尽管使用 GridView 控件可以非常方便地显示、更新和删除数据，但是遗憾的是，GridView 控件不支持数据插入功能，而 DetailsView 控件和 FormView 控件有效地弥补了这一缺陷，有兴趣的读者可以参阅相关资源。本节将介绍如何以编程方式完成数据添加操作，大致需要以下关键步骤：

（1）从界面提取数据。

（2）调用 InsertOne 方法。

（3）操作执行结果提示和异常处理。



添加通讯录数据的代码如下所示：

```
protected void btnAdd_Click(object sender, EventArgs e)
{
    #region 构造通讯录对象
    string Name = tbxName.Text.Trim();
    string Phone = tbxTel.Text.Trim();
    string Email = tbxEmail.Text.Trim();
    string GroupName = dplGroup.SelectedItem.Text;
    #endregion

    //定义数据访问对象
    ContactsDA conDA = new ContactsDA();

    //执行数据添加操作
    try
    {
        conDA.ContactsInsertOne(Name, Phone, Email, GroupName);
        Response.Write("<script>alert('通讯录数据添加成功!')</script>");
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('通讯录数据添加失败。')</script>");
    }
}
```

针对经典的数据访问技术的探讨就到这里，.NET3.0 及其以后的版本引入了一个访问和管理数据的新方法，即语言集成查询（Language Integrated Query，LINQ），由于篇幅所限，此处不再叙述。

6.5　创建外观一致的 Web 站点

专业 Web 应用程序所面临的问题会大大超出单个 Web 页面，因为会涉及更多的 Web 页面及页面之间的外观和内容上的相关性、一致性等。本节将介绍如何在工具的帮助下使用主题和母版页把众多 Web 页面整合成一个完整、统一的网站。

6.5.1　主题与外观

通过前面的介绍，领略了 CSS 样式的优点。但问题是 CSS 规则只限于一组固定的样式特性，它们允许开发人员重用特定的格式化细节（字体、边框、前景和背景色等），但它们显然不能控制 ASP.NET 控件的其他方面，例如，CheckBoxList 控件有一些用于控制如何把

项目组织为行或列的属性，虽然这些属性影响的是控件的外观，但它们在样式的范围之外，所以必须手工设置它们。此外，开发人员可能还希望在定义控件格式的同时定义控件的部分行为，例如，可能希望标准化日历控件的选择模式或者文本框的折行，显然，它们都不可能通过 CSS 实现。

为了解决上述问题，从 ASP.NET 2.0 开始引入了主题（Theme）功能。所谓“主题”是指页面和控件外观属性设置的集合，开发人员利用它不仅能够定义页面和控件的外观，还可以在单个 Web 页面中、单个 Web 应用程序的所有页面甚至所有 Web 应用程序中快速一致地应用所定义的外观。

1. 主题的组成元素

主题由一组文件组成，包括至少一个外观（Skin）文件、若干级联样式表（CSS）文件、若干图片或其他资源。

（1）外观文件

外观文件是主题的核心内容，用于定义页面中服务器控件的外观，一个主题可以包含一到多个皮肤文件，这类文件的扩展名为.skin，可以包含各种服务器控件希望标准化的属性设置。例如：

```
<asp:TextBox runat="server" BorderColor="red" BackColor="white" BorderStyle="dotted" />
```

说明： 上述代码与服务器控件声明代码类似，但 ID 特性不允许在主题中使用，另外，“runat = "server"”部分是必需的，其他所有部分都是可选的。

如果想定义一些特殊用途的控件外观，也可以定义已命名外观（Named Skin），即在控件外观标记中设置 SkinId 属性。

（2）CSS 文件

主题中包含的 CSS 文件与上面介绍的 CSS 文件没有本质区别，不同之处在于使用方式有些变化：一是不需要在页面中指定所用 CSS 文件链接，而是随着设置的主题自动应用相关样式；二是尽量不用 CSS 定义服务器控件的样式，而是处理普通 HTML 控件和页面标记，服务器控件的样式在外观文件中定义。

（3）图片和其他资源

主题还可以包含图形、图像和其他资源，如脚本文件或声音文件。例如，页面主题的一部分可能包括 TreeView 控件的外观。可以在主题中包括用于表示展开按钮和折叠按钮的图形。

2. 创建基本主题

所有的主题都是应用程序相关的。要在 Web 应用程序中使用主题，必须创建一个定义它的文件夹，这个文件夹必须放在一个称为 App_Themes 的文件夹中，而它又必须位于 Web 应用程序的最上层（如图 6-32 所示）。应用程序可以定义多个主题，只要每个主题都在一个单独的文件夹里。对于一个给定的页面，每次只能有一个主题处于活动状态。



接下来需要往主题文件夹中添加外观文件，方法是：在解决方案资源管理器中右击主题的名称，然后在弹出的快捷菜单中选择“添加新项”命令。在弹出的“添加新项”对话框中选择“外观文件”选项，然后在“名称”文本框中输入 skin 文件的名称，最后单击“添加”按钮即可，如图 6-33 所示。

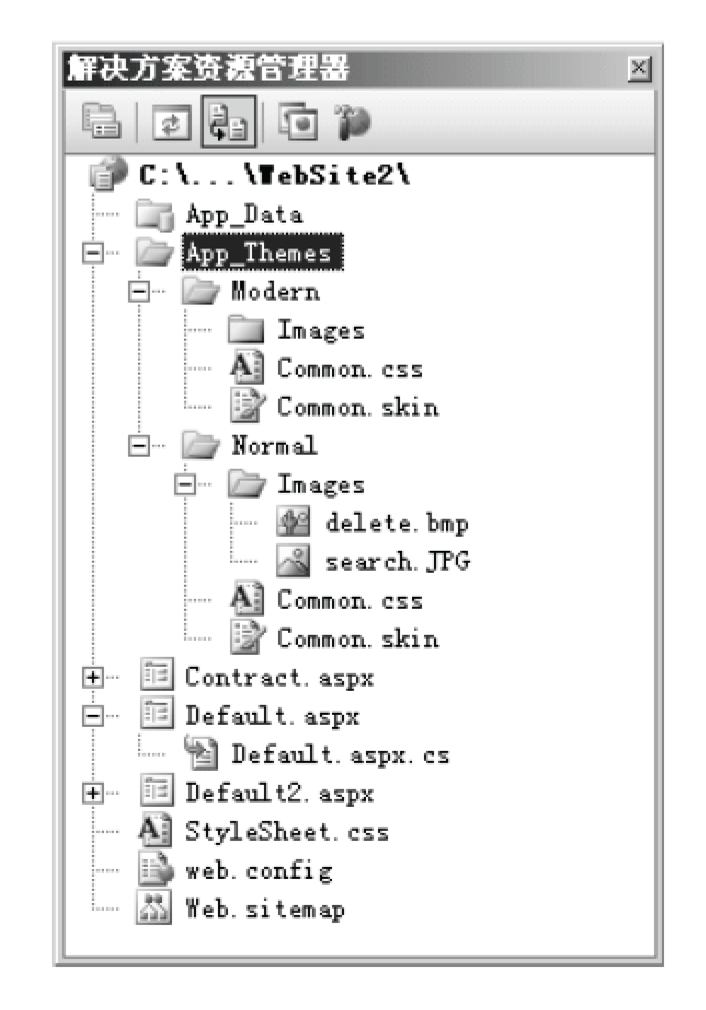


图 6-32　主题文件夹



图 6-33　添加外观文件

在 skin 文件中，使用声明性语法添加标准控件定义，但仅包含要为主题设置的属性。控件定义必须包含“runat="server"”属性，但不能包含 ID=""属性。下列代码显示了外观文件的内容。

```
<asp:Button runat="server" BackColor="#669999"
    BorderColor="Silver" BorderStyle="Solid" BorderWidth="2px"
Font-Bold="True" Font-Size="Medium" ForeColor="Red" />
<asp:TextBox runat="server" BackColor="#CCCCCC"
    BorderStyle="Solid" Font-Bold="True" Font-Size="Large"
    Width="200px"></asp:TextBox>
<asp:TextBox runat="server" BackColor="#CCCCCC"
    BorderStyle="Solid" Font-Bold="True" Font-Size="Large" Height="50px"
    Width="200px"   SkinId="RemarkTextBox" ></asp:TextBox>
```

3．创建包含 CSS 和图像的主题

要在主题中使用样式表，首先必须把样式表放入主题文件夹。ASP.NET 在这个文件夹中搜索所有的 CSS 文件并把它们动态地绑定到所有使用该主题的页面。具体添加和定义样式表的方法见本章前面内容。

外观的另一个强大的功能是可以把图片作为主题的一部分从而重用它们。例如，想把一幅图片用于整个网站的“搜索”按钮或用于所有“删除”按钮，图片文件应该存储在主题文件夹的图片子文件夹里，然后控件外观标记中就可以引用它们了。这里需要强调两点：一是这类控件外观必须是已命名的外观，因为这样定义的标准化按钮类型应该只在需要时

才被页面使用，而不是在定义一个作用于所有按钮的默认样式；二是在外观文件中添加图片的引用时，一定要保证图片的 URL 相对于主题文件夹而不是页面所在的文件夹。主题应用到控件时，ASP.NET 自动在 URL 开始处插入 Themes\ThemeName。下列代码显示了在 Normal 主题中使用图片的外观文件的内容。

```
<asp:ImageButton   runat="server"   SkinId="SearchButton" ImageUrl="Images/search.JPG" />
```

4．应用主题

要让一个主题对 Web 页面起作用，需要在 Page 指令内把 Theme 特性指定为主题所在的文件夹名称，程序代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default"
Theme="Normal"%>
```

把某个主题应用到页面后，ASP.NET 会考虑 Web 页面上的每个控件并检查外观文件中是否为控件定义了属性，如果 ASP.NET 在外观文件中发现了匹配的标签，从外观文件获得的信息就会覆盖控件的当前属性。

图 6-34 显示了一个简单页面应用 Normal 主题后的结果。第一幅图片显示 Default.aspx 页面的原始状态，它没有使用主题；第二幅图片显示应用 Normal 后的同一个页面。Normal 中的所有设置被应用到 Default.aspx 控件中，即它们会覆写某些已在页面上显式设置的值（如文本框的背景色）。不过，原始页面中且和主题没有冲突的细节（如为按钮的字体类型）得以保留。

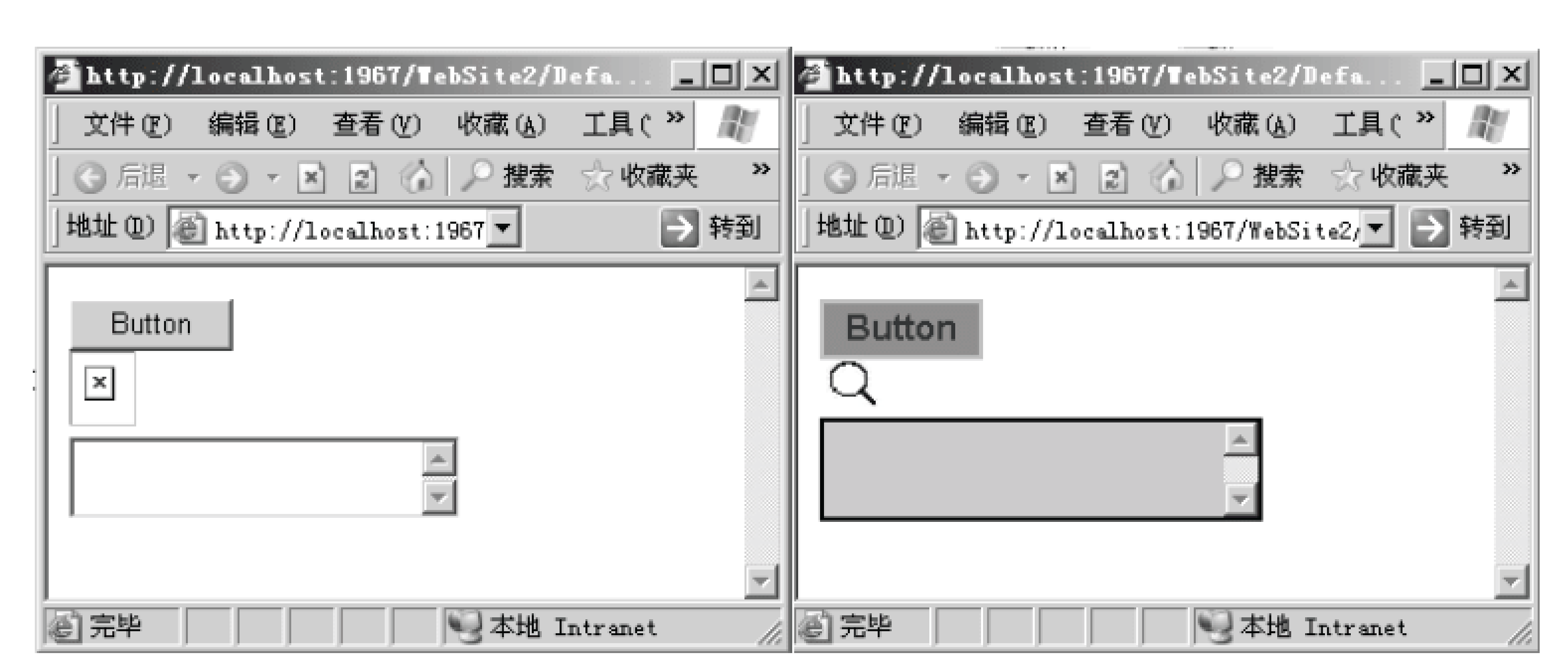


图 6-34　应用主题

如果不想对页面上的某个控件应用主题，则可以将该控件的 EnableTheming 属性设置为 false。

5．其他

如果希望把主题应用到整个 Web 应用程序，最简捷的办法是在 web.confg 文件的<pages>元素中为应用程序配置主题。或者，如果希望让用户在程序运行阶段选择主题，则需要以编程方式动态设置 Page.Theme 属性或 Page.StyleSheet 属性。

由于篇幅所限，此处不再对上述内容展开讨论，感兴趣的读者可以参阅相关文献。



6.5.2 用母版页统一页面布局

为了给访问者一致的感受，标准化网站的格式只是整个过程的一部分，还要保证有同样的元素，如网站的标题、导航控件等在每个页面中出现在相同的位置，如图 6-22 所示。将这些公共元素在每个页面中多次复制不是解决问题的理想方法，同时会带来难以维护等弊端。解决这一问题的关键在于创建一个可以重复应用到整个网站的简单而灵活的布局。

母版页（MasterPage）是 ASP.NET 的一个创新，它专门设计用于标准化的 Web 页面布局，它允许开发人员创建可重用的页面模板。使用母版页可以为网站的页面定义布局，包括所有常用的细节，如页头、菜单栏、广告条。规范化结构后，就可以在整个网站使用母版页，从而保证所有的页面都具有相同的设计。访问者从一个部分跳转到其他部分时也不会感觉布局发生了显著的变化，从而改善了用户体验。

在实现网站一致性的过程中，必须包含两种文件：一种是母版页，另一种是内容页。母版页后缀名是.master，它封装页面中的公共元素；内容页实际是普通的.aspx 文件，它包含除母版页之外的其他非公共内容。在运行过程中，ASP.NET 引擎将两种页面内容合并执行，最后将结果发给客户端浏览器。

虽然母版页和内容页功能强大，但是其创建和应用过程并不复杂。下面介绍在 VWD 2008 中母版页的简单用法。

1. 创建母版页

母版页中包含的是页面公共部分，即网页模板，因此，在创建示例之前，必须分析页面结构，判断哪些内容是页面公共部分。

创建母版页的方法如下：

在解决方案资源管理器中右击网站的名称，然后在弹出的快捷菜单中选择“添加新项”命令。在“添加新项”对话框中选择“母版页”选项，然后在“名称”文本框中输入“MyMaster”，在“语言”下拉列表框中选择想使用的编程语言，然后单击“添加”按钮，即会在“源”视图中打开新的母版页，程序如下：

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MyMasterPage.master.cs" Inherits="MyMasterPage" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/ TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <asp:ContentPlaceHolder id="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
```

```
        <div>
            <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

            </asp:ContentPlaceHolder>
        </div>
        </form>
</body>
</html>
```

以上母版页代码与普通.aspx 文件代码存在一些相似之处，但更重要的是存在 3 点差异：一是母版页的扩展名是.master，客户端浏览器不能直接访问母版页；二是普通.aspx 文件的代码头声明是<%@ Page %>，而母版页文件的代码头必须声明为<%@ Master %>；三是母版页中可以包括一个或者多个 ContentPlaceHolder 控件，而在普通.aspx 文件中是不包含该控件的。ContentPlaceHolder 控件起到一个占位符的作用，能够在母版页中标识出某个区域，该区域将用内容页中的特定代码代替。

创建母版页后，就需要在其中添加代表网页公共部分的元素，如静态文本和控件的任何组合。

2. 创建内容页

内容页主要包含页面中的非公共内容。

内容页的创建方法与添加普通 Web 页面类似，不同的是需要在“添加新项”对话框中选中“选择母版页”复选框（如图 6-35 所示），然后单击“添加”按钮，打开“选择母版页”对话框，如图 6-36 所示，选择 MyMasterPage.master 选项，然后单击“确定”按钮，即创建一个新的.aspx 文件。

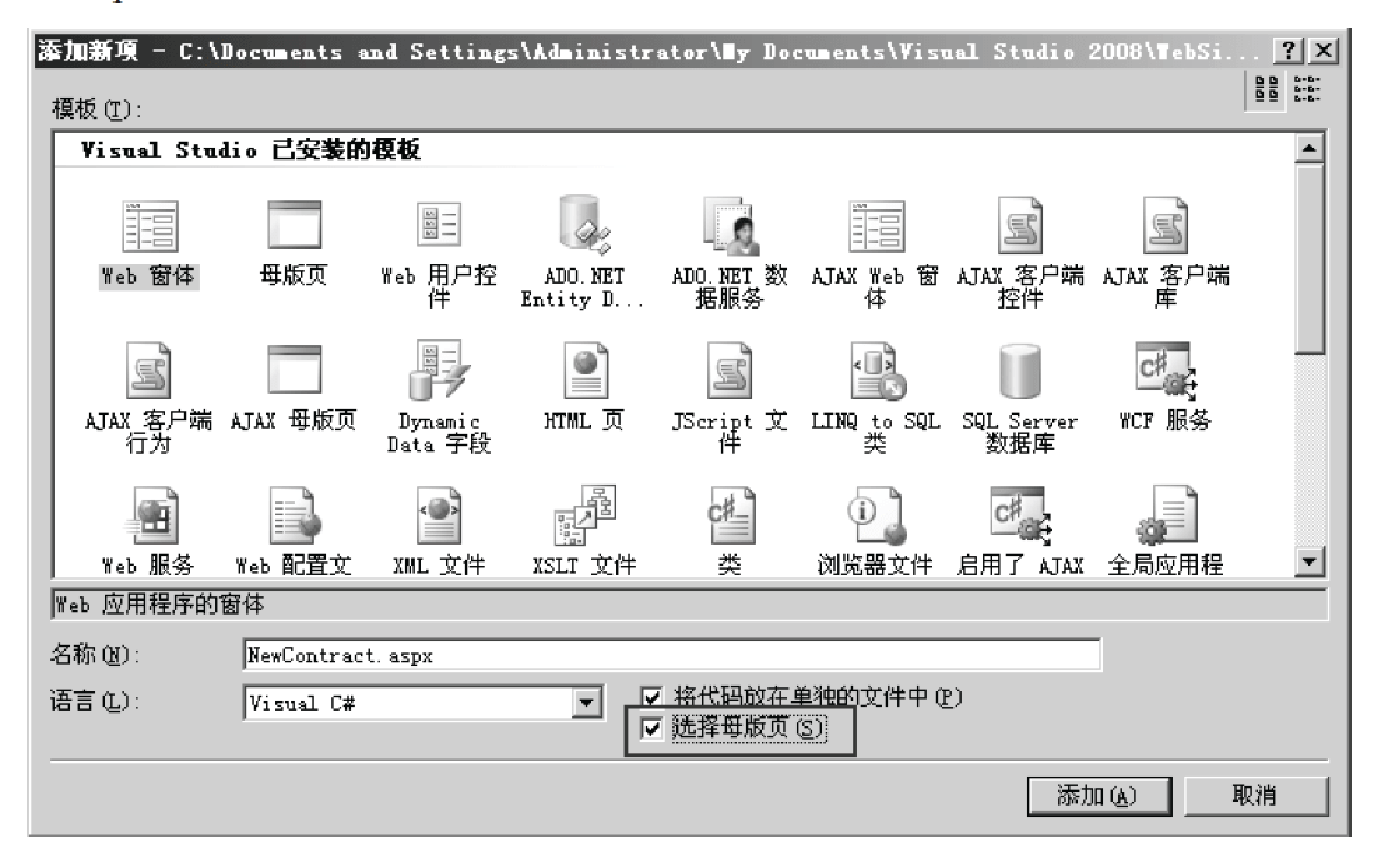


图 6-35　添加内容页





图 6-36　选择母版页

虽然内容页的扩展名是.aspx，但是，其代码结构与普通.aspx 文件有着很大差异。一般内容页的代码结构如下：

```
<%@ Page Title="" Language="C#" MasterPageFile="~/MyMasterPage.master" AutoEventWireup=
"true" CodeFile="NewContact.aspx.cs" Inherits="NewContact" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
</asp:Content>
```

由以上代码可知，内容页的代码主要分为两个部分：代码头声明和 Content 控件。内容页的代码头声明与普通.aspx 文件很相似，只是增加了属性 MasterPageFile 和 Title 设置。属性 MasterPageFile 用于设置该内容页所绑定的母版页的路径，属性 Title 用于设置页面 title 属性值。另外，在内容页中，还可以包括一个或者多个 Content 控件。页面中所有非公共内容都必须包含在 Content 控件中。每一个 Content 控件通过属性 ContentPlaceHolderID 与母版页中的 ContentPlaceHolder 控件相连接。通过以上设置，就可以实现母版页与内容页的绑定。

在图 6-22 中的解决方案文件中添加母版页 MyMaterPage.master，将页面 Contact.aspx 中的外层 HTML 表复制到母版页，同时把表的中间右侧部分替换为 ContentPlaceHolder 控件，如图 6-37 所示。然后添加使用该母版页的内容页 NewContact.aspx，如图 6-38 所示，母版页元素在内容页上是不可编辑的，开发人员唯一可以操作的是 ContentPlaceHolder 控件所占区域。从工具箱拖拽一个 div 标记到 ContentPlaceHolder 控件中，并将图 6-22 中 Contact.aspx 页面中间右侧部分复制到 div 标记内，如图 6-39 所示。

通过对比图 6-22 和图 6-39，能够发现母版页和内容页结合得多么完美，并且其他页面也可以套用该母版页，从而实现网站布局的高度一致性。

3. 其他

有关母版页的内容还有很多，如嵌套母版页、动态加载母版页、编程访问母版页控件等，感兴趣的读者可参阅其他相关资料。

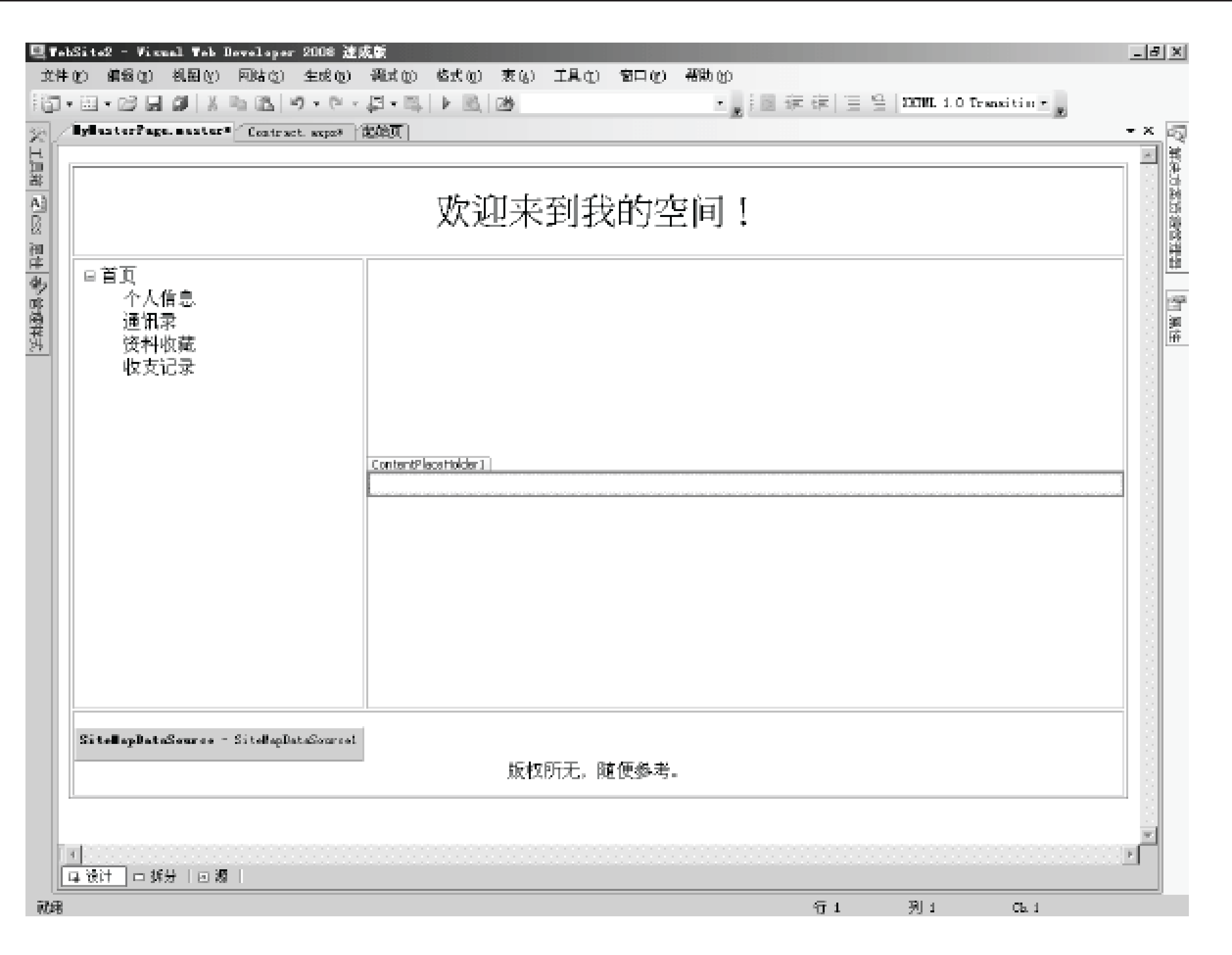


图 6-37　设计母版页

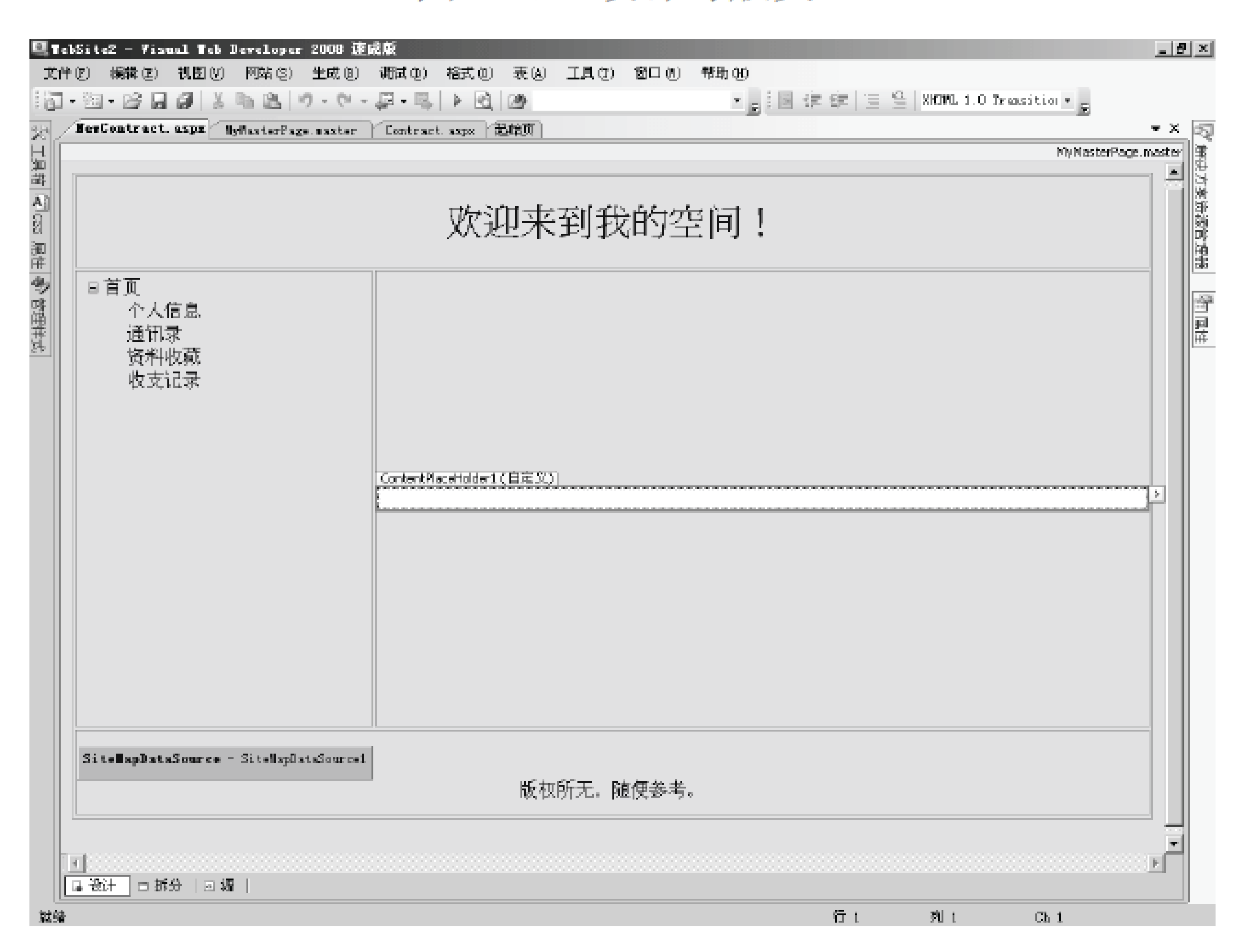


图 6-38　设计内容页

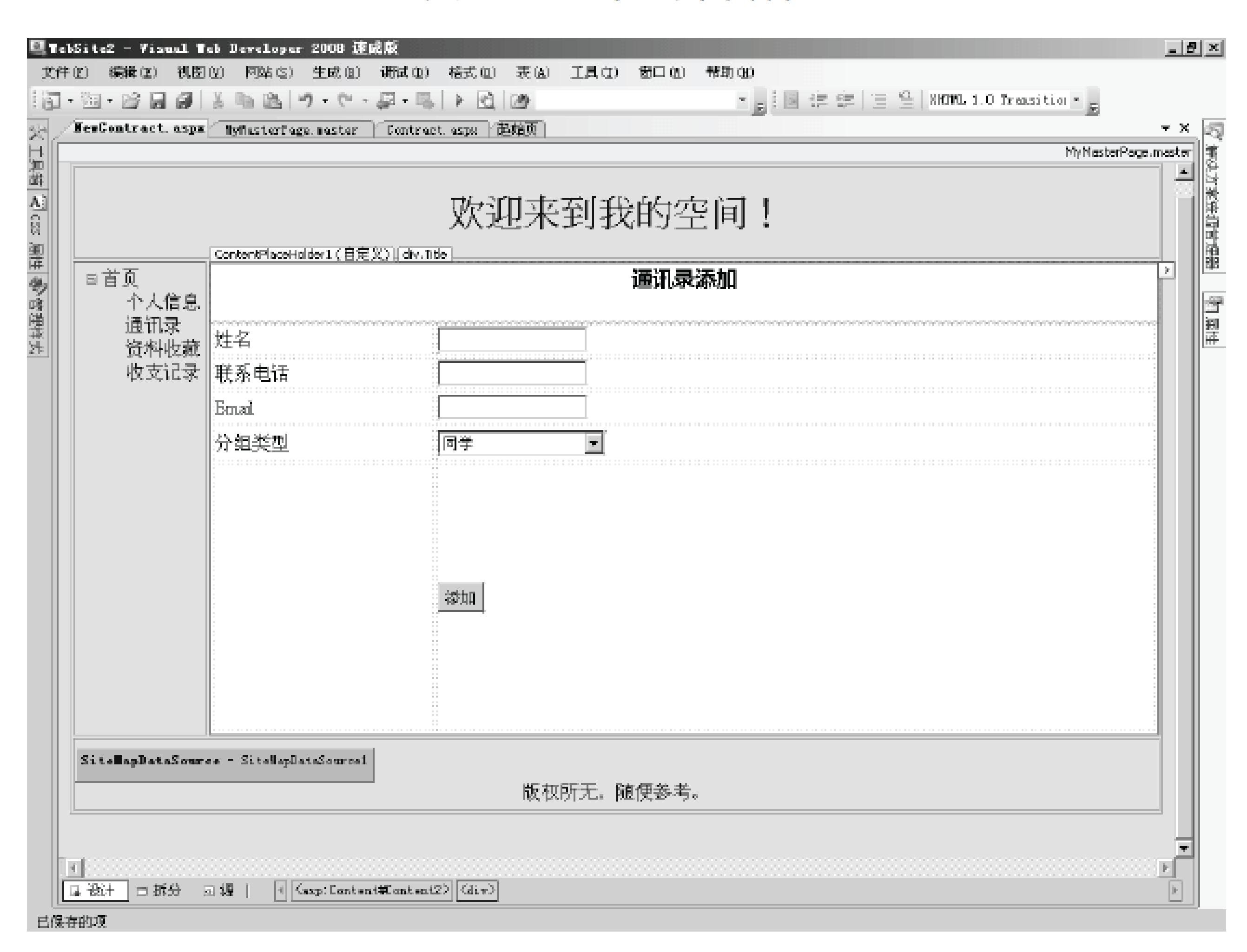


图 6-39　布局内容页



6.6　ASP.NET 内置对象及应用

ASP.NET 提供了许多内置对象，前面所使用的 Response 对象就是其中一个，另外还包括 page 对象、request 对象、server 对象、application 对象、session 对象、cookie 对象和 cache 对象等。这些对象提供了相当多的功能，例如，可以在两个网页之间传递变量、输出数据以及记录变量值等。因为是内置对象，开发人员可以在应用程序中直接引用这些对象来实现特定的功能。本节首先简要介绍常用内置对象，然后综合介绍它们的应用。

6.6.1　常用内置对象简介

1．response 对象

response 对象可以输出信息到客户端，包括直接发送信息给浏览器、重定向浏览器到另一个 URL 或设置 cookie 的值。它是 HttpResponse 类的一个实例，该类主要封装来自 ASP.NET 操作的 HTTP 响应信息。表 6-6 和表 6-7 分别列举了 response 对象几个常用的属性和方法。

表 6-6　response 对象的属性

属　　性	说　　明	属　性　值
BufferOutput	获取或设置一个值，该值指示是否缓冲输出，并在处理完成整个页后将其发送	如果缓冲到了客户端的输出，则为 true；否则为 false。默认为 true
Cache	获取 Web 页的缓存策略（过期时间、保密性、变化子句）	包含有关当前响应的缓存策略信息的 HttpCachePolicy 对象
Charset	获取或设置输出流的 HTTP 字符集	输出流的 HTTP 字符集
IsClientConnected	获取一个值，通过该值指示客户端是否仍连接在服务器上	如果客户端当前仍在连接，则为 true；否则为 false

表 6-7　response 对象的方法

方　　法	说　　明
Write	将指定的字符串或表达式的结果写到当前的 HTTP 输出
End	停止页面的执行并得到相应结果
Clear	用来在不将缓存中的内容输出的前提下，清空当前页的缓存，仅当使用了缓存输出时，才可以利用 Clear 方法
Flush	将缓存中的内容立即显示出来。该方法有一点和 Clear 方法一样，它在脚本前面没有将 Buffer 属性设置为 true 时会出错。和 End 方法不同的是，该方法调用后，该页面可继续执行
Redirect	使浏览器立即重定向到程序指定的 URL

2. request 对象

request 对象能够读取客户端在 Web 请求期间发送的 HTTP 值，它是 HttpRequest 类的一个实例。表 6-8 和表 6-9 分别列举了 request 对象几个常用的属性和方法。

表 6-8　request 对象的属性

属　　性	说　　明	属　性　值
QueryString	获取 HTTP 查询字符串变量集合	NameValueCollection 对象
Path	获取当前请求的虚拟路径	当前请求的虚拟路径
UserHostAddress	获取远程客户端的 IP 主机地址	远程客户端的 IP 地址
Browser	获取有关正在请求的客户端的浏览器功能的信息	HttpBrowserCapabilities 对象

表 6-9　request 对象的方法

方　　法	说　　明
BinaryRead	执行对当前输入流进行指定字节数的二进制读取
MapPath	为当前请求将请求的 URL 中的虚拟路径映射到服务器上的物理路径

3. application 对象

application 对象是 HttpApplicationState 类的一个实例。

HttpApplicationState 类的单个实例，将在客户端第一次从某个特定的 ASP.NET 应用程序虚拟目录中请求任何 URL 资源时创建。对于 Web 服务器上的每个 ASP.NET 应用程序，都要创建一个单独的实例，然后通过内部 application 对象公开对每个实例的引用。

application 对象使给定应用程序的所有用户之间共享信息，并且在服务器运行期间持久地保存数据。表 6-10 和表 6-11 分别列举了 application 对象几个常用的属性和方法。

表 6-10　application 对象的属性

属　　性	说　　明	属　性　值
AllKeys	获取 HttpApplicationState 集合中的访问键	HttpApplicationState 对象名的字符串数组
Count	获取 HttpApplicationState 集合中的对象数	集合中的 Item 对象数，默认为 0

表 6-11　application 对象的方法

方　　法	说　　明
Add	新增一个新的 application 对象变量
Clear	清除全部的 application 对象变量
Get	使用索引关键字或变数名称得到变量值
GetKey	使用索引关键字来获取变量名称
Lock	锁定全部的 application 变量
Remove	使用变量名称删除一个 application 对象
RemoveAll	删除全部的 application 对象变量
Set	使用变量名更新一个 application 对象变量的内容
UnLock	解除锁定的 application 变量



4．session 对象

session 对象是 HttpSessionState 类的一个实例，该类为当前用户会话提供信息，还提供对可用于存储信息的会话范围的缓存的访问以及控制如何管理会话的方法。

在应用程序中，可以使用 session 对象存储特定用户会话所需的信息。这样，当用户在应用程序的不同 Web 页之间跳转时，存储在 session 对象中的变量值将不会丢失，而是在整个用户会话中一直存在下去，直到超出其生存有效期。

需要强调的是，session 对象是与特定用户相联系的，针对某一个用户赋值的 session 对象是和其他用户的 session 对象完全独立的，不会相互影响。表 6-12 和表 6-13 分别列举了 session 对象几个常用的属性和方法。

表 6-12　session 对象的属性

属　　性	说　　明	属　性　值
Count	获取会话状态集合中 session 对象的个数	session 对象的个数
TimeOut	获取并设置在会话状态提供程序终止会话之前各请求之间所允许的超时期限	超时期限（以分钟为单位）
SessionID	获取用于标识会话的唯一会话 ID	会话 ID

表 6-13　session 对象的方法

方　　法	说　　明
Add	新增一个 session 对象
Clear	清除会话状态中的所有值
Remove	删除会话状态集合中的项
RemoveAll	清除所有会话状态值

5．server 对象

server 对象提供对服务器上的方法和属性的访问，它是 HttpServerUtility 的一个实例。表 6-14 和表 6-15 分别列举了 server 对象几个常用的属性和方法。

表 6-14　server 对象的属性

属　　性	说　　明	属　性　值
MachineName	获取服务器的计算机名称	本地计算机的名称
ScriptTimeout	获取和设置请求超时	请求的超时设置（以秒计）

表 6-15　server 对象的方法

方　　法	说　　明
CreateObject	创建 COM 对象的一个服务器实例
CreateObjectFromClsid	创建 COM 对象的服务器实例，该对象由对象的类标识符（CLSID）标识

续表

方　法	说　明
Execute	使用另一页执行当前请求
Transfer	终止当前页的执行，并为当前请求开始执行新页
HtmlDecode	对已被编码以消除无效 HTML 字符的字符串进行解码
HtmlEncode	对要在浏览器中显示的字符串进行编码
MapPath	返回与 Web 服务器上的指定虚拟路径相对应的物理文件路径
UrlDecode	对字符串进行解码，该字符串为了进行 HTTP 传输而进行编码并在 URL 中发送到服务器
UrlEncode	编码字符串，以便通过 URL 从 Web 服务器到客户端进行可靠的 HTTP 传输

6．cookie 对象

cookie 与 session、application 类似，也是用来保存相关信息的，但 cookie 和其他对象的最大不同是，cookie 将信息保存在客户端，而 session 和 application 将信息保存在服务器端。表 6-16 和表 6-17 分别列举了 cookie 对象几个常用的属性和方法。

表 6-16　cookie 对象的属性

属　性	说　明	属　性　值
Name	获取或设置 cookie 的名称	cookie 的名称
Value	获取或设置 cookie 的 Value	cookie 的 Value
Expires	获取或设置 cookie 的过期日期和时间	作为 DateTime 实例的 cookie 过期日期和时间
Version	获取或设置此 cookie 符合的 HTTP 状态维护版本	此 cookie 符合的 HTTP 状态维护版本

表 6-17　cookie 对象的方法

方　法	说　明
Add	新增一个 cookie 变量
Clear	清除 cookie 集合内的变量
Get	通过变量名或索引得到 cookie 的变量值
GetKey	以索引值来获取 cookie 的变量名称
Remove	通过 cookie 变量名来删除 cookie 变量

ASP.NET 包含两个内部 cookie 集合：一是通过 HttpRequest 的 cookies 集合访问的集合包含通过 cookie 标头从客户端传送到服务器的 Cookie；二是通过 HttpResponse 的 cookies 集合访问的集合包含一些新 cookie，这些 cookie 在服务器上创建并以 Set-Cookie 标头的形式传输到客户端。

使用 cookie 对象具有的优点是可配置到期规则、不需要任何服务器资源、简单性和数据持久性。但是，使用 cookie 对象带来的缺点也不容忽视，如空间大小受到限制、容易被用户禁用和潜在的被篡改的风险等。



7．cache 对象

对于每个应用程序域均创建该类的一个实例，并且只要对应的应用程序域保持活动，该实例便保持有效。有关此类实例的信息通过 HttpContext 对象的 cache 属性或 page 对象的 cache 属性来提供。表 6-18 和表 6-19 分别列举了 cache 对象几个常用的属性和方法。

表 6-18　cache 对象的属性

属　　性	说　　明	属　性　值
Count	获取存储在缓存中的项数。当监视应用程序性能或使用 ASP.NET 跟踪功能时，此属性可能非常有用	存储在缓存中的项数
Item	获取或设置指定键的缓存项	表示缓存项的键的 string 对象

表 6-19　cache 对象的方法

方　　法	说　　明
Add	将指定项添加到 cache 对象，该对象具有依赖项、过期和优先级策略以及一个委托（可用于在从 cache 移除插入项时通知应用程序）
Get	从 cache 对象检索指定项
Remove	从应用程序的 cache 对象移除指定项
Insert	向 cache 对象插入项。使用此方法的某一版本改写具有相同 key 参数的现有 Cache 项

8．viewState 对象

Web 窗体是无状态的，viewState 是.NET 中提出的状态保存的一种新途径，引入 viewState 的一个主要原因是，session 值是保存在服务器内存上，大量地使用 session 将导致服务器负担加重。而 viewState 由于只是将数据存入到页面隐藏控件中，不再占用服务器资源，因此，可以将一些需要服务器“记住”的变量和对象保存到 viewState 中。而 session 则只应该应用在需要跨页面且与每个访问用户相关的变量和对象存储上。但 viewState 并不是能存储所有的.NET 类型数据，它仅支持 String、Integer、Boolean、Array、ArrayList、Hashtable 以及自定义的一些类型。它系统地实现了保存控件状态的功能，服务器端控件能够在多次请求之间保存状态也全靠它。

6.6.2　内置对象的综合应用举例

前面内容将通讯录的添加和显示集中在一个页面上，貌似效果还不错，但是，在数据项比较多的情况下，这样的布局会使界面及代码过于繁琐，且不易维护。本节将对其从以下几个方面加以改进。

（1）把添加和显示功能分开在两个页面实现，当单击某一行记录时，网站导航到详细内容页面上，并显示被单击的记录。主要步骤如下：

① 在维护页面中给 GridView1 控件添加一个 ButtonField 列，其 Text 属性值设置为“详细”，CommandName 属性值设置为 Details，其他属性按默认值处理，如图 6-40 所示。

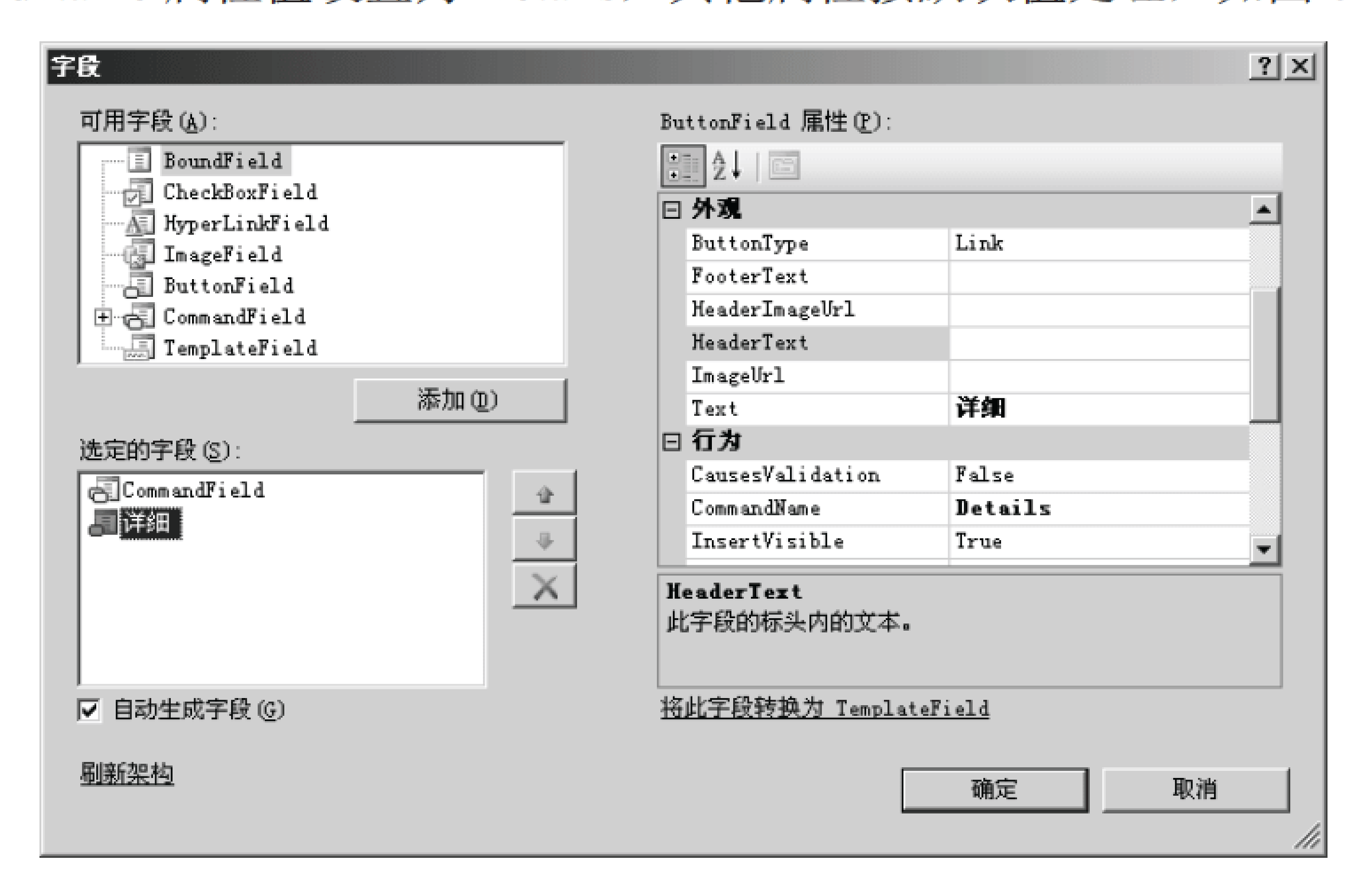


图 6-40　添加 ButtonField 字段

② 为 GridView1 控件的 RowCommand 事件添加处理代码，当单击“详细”按钮时，首先获取被点记录所在行序号，然后提取对应的主键字段值，并存储在 session 对象中，最后跳到 Contract.aspx 页面。

```
protected void GridView1_RowCommand(object sender, GridViewCommandEventArgs e)
{
    int index = Convert.ToInt32(e.CommandArgument);
    int SN = 0;
    try
    {
        switch (e.CommandName.Trim())
        {
            case "Details":
                SN = (int)(GridView1.DataKeys[index].Value);
                Session["ContactSN"] = SN;
                Response.Redirect("~/Contract.aspx");
                break;
            default:
                break;
        }
    }
    catch (Exception Ex)
    {
        Response.Write("<script>alert('系统出现错误，请稍后重试。" + Ex.Message + "')</script>");
    }
}
```



③ 在 Contacts.aspx 页面代码中为 Page_Load 事件处理方法添加如下代码，通过判断 Session["ContactSN"]是否为空来决定是否有对象数据需要在页面上显示。

```
protected void Page_Load(object sender, EventArgs e)
{
    if (Session["ContactSN"] != null)
    {
        int SN = (int)Session["ContactSN"];
        Session["ContactSN"] = null;//清空 Session
        ViewState["ContactSN"] = SN;//使用 ViewState 保存记录号

        #region 绑定数据到界面
        tbxName.Text = conMod.Name;
        tbxTel.Text = conMod.Phone;
        tbxEmail.Text = conMod.Email;
        dplGroup.Text = conMod.GroupName;
        #endregion
    }
}
```

④ 为了能够在添加操作和维护操作两个模块之间轻松切换，还需修改站点地图文件，代码如下：

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
    <siteMapNode url="" title="首页"  description="">
      <siteMapNode url="" title="通讯录"  description="" >
        <siteMapNode url="~/Contract.aspx" title="通讯录添加"  description="" />
        <siteMapNode url="~/ContactsMaintain.aspx" title="通讯录维护"  description="" />
      </siteMapNode>
    </siteMapNode>
</siteMap>
```

（2）当数据项很多时，直接在 GridView 控件中编辑数据不够方便，所以，还需要一个页面既能显示数据又能保存编辑后的结果。虽然仿照添加模块再做一个修改模块也不是难事，但是考虑到代码的复用，将修改操作和添加操作放在一个页面上是一个不错的方案。主要步骤如下：

将“添加”按钮名称改为“保存”，同时，将按钮的 Click 事件处理方法代码修改如下：

```
protected void btnAdd_Click(object sender, EventArgs e)
{
    #region 构造通讯录对象
    conMod.Name = tbxName.Text.Trim();
    conMod.Phone = tbxTel.Text.Trim();
    conMod.Email = tbxEmail.Text.Trim();
    conMod.GroupName = dplGroup.SelectedItem.Text;
```

```
        #endregion

        //执行数据添加操作
        try
        {
            if (ViewState["ContactSN"] == null)
            {
                conDA.ContactsInsertOne(conMod);
                Response.Write("<script>alert('通讯录数据添加成功!')</script>");
            }
            else
            {
                conMod.SN = (int)ViewState["ContactSN"];
                conDA.ContactsUpdateOne(conMod);
                Response.Write("<script>alert('通讯录数据修改成功!')</script>");
                ViewState["ContactSN"] = null;//清空
            }
        }
        catch (Exception ex)
        {
            Response.Write("<script>alert('通讯录数据添加失败。')</script>");
        }
    }
```

（3）前面代码中的页面跳转语句也可以使用如下代码代替。请读者仔细观察两者的不同，并分析原因。

```
Server.Transfer("~/Contract.aspx");
```

（4）传递简单数值时也可以使用地址参数来实现，需要在上述代码基础上做如下工作。

① 在页面跳转时，采用如下代码取代 Session 对象：

```
Response.Redirect("~/Contract.aspx?sn="+SN.ToString());
```

② 在目标页面中，使用 Request 对象提取参数值，代码如下：

```
int SN = Int32.Parse(Request.QueryString["sn"]);
```

由于篇幅所限，有关 cookie 对象、application 对象和 cache 对象的使用方法在此处就不再介绍了，感兴趣的读者可以参阅相关资料做进一步学习。

6.7 小　　结

本章介绍了开发 Web 应用程序的一种技术，即 ASP.NET，并针对 ASP.NET 3.5 讨论了



如何在 Web 页面中使用 CSS、如何使用服务器控件、如何创建外观一致的 Web 页面、如何处理数据库中的数据、如何使用常用内置对象等几个方面的内容，一定程度上体现了 ASP.NET 在 Web 应用开发的特点。但是，必须要强调的一点是，ASP.NET 的特性并不止这几个方面，本章介绍的内容相对于整个 ASP.NET 技术只是其中的一部分（由于篇幅有限）。读者要想进一步提高，使用 ASP.NET 技术开发健壮、安全、用户体验良好、数据库驱动的 Web 应用程序的能力，请进一步阅读其他相关书籍和资料。

6.8　思　考　题

1．按定义位置，CSS 样式可以分为哪些类型？

2．利用 VWD 2008 提供的 CSS 相关工具生成如下样式，并存放在外部样式表中，最后将该样式应用到 Web 页面中。

```
*
{
font-family: Arial;
}

h1
{
font-size: 20px;
}

#MainContent
{
width: 644px;
float: left;
}

#Sidebar
{
background-color: Gray;
width: 200px;
float: left;
}

#Footer
{
background-color: #C0C0C0;
width: 844px;
clear: both;
}
```

3．ASP.NET 服务器控件的 Runat="Server"应该怎么理解？

4．试简要说明服务器控件在 ASP.NET 应用程序页面中的重要作用。

5．试改造通讯录添加的事件处理代码，以调用接收实体对象参数的 ContactsInsertOne 方法版本。

6．自己设计一个母版页，并创建若干内容页与之关联，形成布局风格一致的个人信息管理系统。

7．试说明母版页、内容页、普通 Web 页面之间的相同点和不同点。

第 7 章　基于 JSP 的服务器端程序设计

JSP（Java Server Pages）是由 Sun 公司开发的一种服务器端脚本语言，自 1999 年推出以来，其逐步发展成为开发 Web 应用系统的一项重要技术。JSP 可以嵌套在 HTML 中，而且支持多个操作系统平台，使用 JSP 开发的 Web 应用系统无须进行较大改动就可以在不同的操作系统中运行。

本章首先介绍 JSP 的工作原理、运行环境，然后介绍 JSP 语法知识、常用内置对象，最后介绍在数据库实际开发中的 JDBC 技术及其方法。

7.1　JSP 简介

随着 Internet 的不断发展，动态网页技术日益流行，JSP 作为主流的动态网页技术之一，其优势日益显著。JSP 是由 Sun Microsystems 公司倡导、许多公司参与一起建立的一种动态网页技术标准，是基于 Java Servlet 以及整个 Java 体系的 Web 开发技术，具有动态页面与静态页面分离、能够脱离硬件平台束缚、一次编写，跨平台运行等优点。利用这一技术可以建立安全、跨平台的先进动态网站。

7.1.1　JSP 的特点

1. 将内容的生成和显示进行分离

使用 JSP 技术，Web 页面开发人员可以使用 HTML 或 XML 标识来设计和格式化最终页面。使用 JSP 标识或脚本来生成页面上的动态内容（内容是根据请求变化的，如请求账户信息或特定的一件商品的价格）。生成内容的逻辑被封装在标识和 JavaBeans 组件中，并且捆绑在脚本中，所有的脚本在服务器端运行。如果核心逻辑被封装在标识和 Beans 中，那么其他人，如 Web 管理人员和页面设计者，就能够编辑和使用 JSP 页面，而不影响内容的生成。

在服务器端，JSP 引擎解释 JSP 标识和脚本，生成所请求的内容（例如，通过访问 JavaBeans 组件，使用 JDBCTM 技术访问数据库，或者包含文件），并且将结果以 HTML（或 XML）页面的形式发送回浏览器。这有助于用户保护自己的代码，同时又保证了任何基于 HTML 的 Web 浏览器的完全可用性。

2．生成可重用的组件

绝大多数 JSP 页面依赖于可重用的、跨平台的组件（JavaBeans 或 Enterprise JavaBeansTM）来执行应用程序所要求的更为复杂的处理。开发人员能够共享和交换执行普通操作的组件，或者使这些组件为更多的使用者或客户团体所使用。基于组件的方法加速了总体开发过程，并且使各种组织在它们现有的技能和优化结果的开发努力中得到平衡。

3．采用标识简化页面开发

并不是所有的 Web 页面开发人员都是熟悉脚本语言的编程人员。Java Server Pages 技术封装了许多功能，这些功能是在易用的、与 JSP 相关的 XML 标识中进行动态内容生成所需要的。标准的 JSP 标识能够访问和实例化 JavaBeans 组件、设置或者检索组件属性、下载 Applet 以及执行用其他方法更难于编码和耗时的功能。

通过开发定制化标识库，JSP 技术是可以扩展的。今后，第三方开发人员和其他人员可以为常用功能创建自己的标识库，这使得 Web 页面开发人员能够使用熟悉的工具和如同标识一样的执行特定功能的构件来工作。

4．将内容的生成和显示进行分离

JSP 能提供所有 Servlets 功能，而且比用 println 书写和修改 HTML 更方便。此外，还可以更明确地进行分工，Web 页面设计人员编写 HTML，只需要留出地方让 Servlets 程序员插入动态部分即可。

5．具有 Java 技术的所有优点

由于 JSP 页面的内置脚本语言是基于 Java 编程语言的，而且所有的 JSP 页面都被编译成为 Java Servlet，JSP 页面就具有 Java 技术的所有优点，包括健壮的存储管理和安全性。

6．作为 Java 平台的一部分，JSP 拥有 Java 编程语言“一次编写，跨平台运行”的特点

随着越来越多的供应商将 JSP 支持添加到它们的产品中，用户可以使用自己所选择的服务器和工具，更改工具或服务器并不影响当前的应用。

7.1.2　JSP 工作原理

JSP 文件的运行是“编译式”，而不是“解释式”，即在 JSP 页面执行时，服务器把 JSP 文件先翻译成 Servlet 形式的 Java 类型的字节码文件，然后通过 Java 虚拟机来运行。所以从本质上来讲，运行 JSP 文件最终还是要通过 Java 虚拟机，但根据 JSP 技术的相关规范，JSP 语言必须在某个构建于 Java 虚拟机之上的特殊环境中运行，这个特殊环境就是 Servlet Container（Servlet 容器），而且每个 JSP 页面在被系统调用之前，必须先被 Servlet 容器解析成一个 Servlet 文件。整个 JSP 的运行流程如图 7-1 所示。



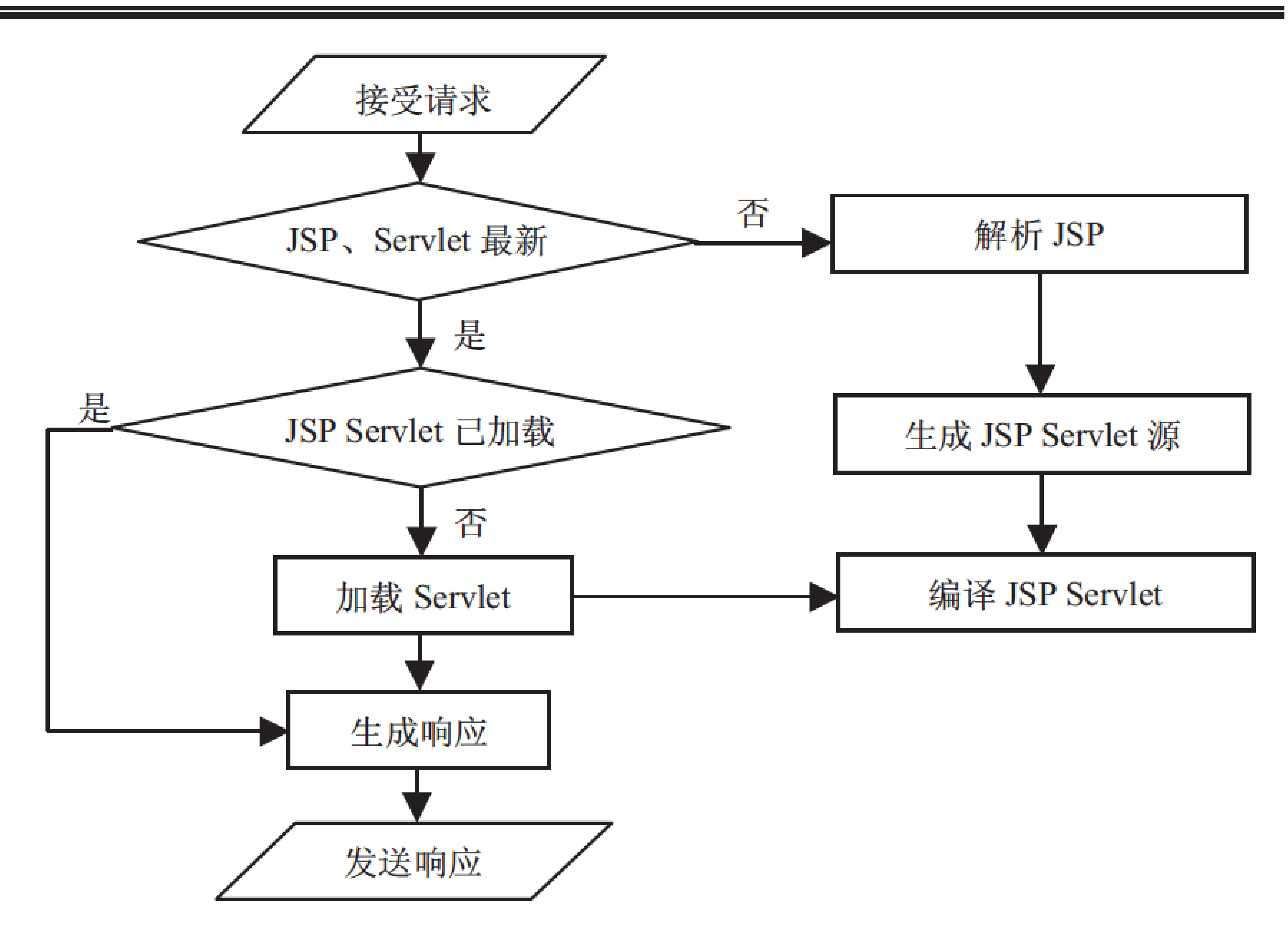


图 7-1　整个 JSP 的运行流程

从 JSP 的工作原理来看，JSP 程序既能以 Java 语言的方式处理 Web 程序中的业务逻辑，也可以处理基于 HTML 协议的请求，它是集众多功能于一身的。

不过，在编写程序的过程中，不能过多地在 JSP 代码中混杂提供显示功能和提供业务逻辑的代码，而是要把 JSP 程序定位到“管理显示逻辑”的角色上。

7.1.3　JSP 的基本语法

JSP 的语法不是特别复杂，一般来讲，在 JSP 类型的程序中，主要包括声明语句、表达式和脚本代码 3 个要素。

1．声明语句

JSP 的声明语句语法格式是<%!declaration;%>，例如：

```
<%! int i=1; %>
<%! int a,b,c; %>
<%! String str="hello"; %>
```

2．表达式

表达式的语法格式为<%=expression%>，例如：

```
<%=new Date().toString() %>
<%=1+2+3 %>
```

3. 脚本代码

JSP 脚本代码即 JSP 文件中的 Java 代码部分，在其中可以使用任何 Java 语法。JSP 脚本代码的语法格式为<%script%>，例如：

```
<%
System.out.println("Hello, World!");
%>
```

脚本代码可以看作是 JSP 的实现业务逻辑部分，正因为有了这些代码，JSP 才可以是 HTML 元素加 Java 语言的混合体。

7.1.4　JSP 和 Java Servlet 技术

早期的动态 Web 技术是基于 CGI（Common Gateway Interface，通用网关接口）的，这种技术使用一些脚本类语言，如 Perl 来获取 Web 请求，完成后台动作，然后将结果返回给用户。但是这种脚本语言被请求都会产生一个后台进程，会严重消耗服务器极大的资源。

Servlet 技术是 Java 动态 Web 技术的基础，它是用 Java 书写的一种规范，是与平台无关的服务器构件，可以在支持 Servlet 的 Web 服务器或应用服务器上运行。Servlet 技术将动态程序编译成字节码，然后在 Web 容器中运行。

Servlet 为客户端和服务器端的信息处理提供了一种“请求/回答”机制。Java 的 Servlet API 为处理客户端和服务器的请求和回答信息定义了标准接口。其过程如下：

（1）客户端向服务器发送请求。

（2）服务器将请求发送至 Servlet。

（3）根据客户端的请求，Servlet 动态构造应答信息并返回给服务器，此时，也可能会利用外部资源。

（4）最后，服务器端将应答返回到客户端。

7.2　JDK 的获取与安装

JSP 的运行环境包括 JDK 和 Web 服务器。下面首先从 Sun 公司官网获取 JDK 的下载地址并下载到本地。

7.2.1　JDK 的下载

下载 JDK 之前，要清楚 Java SE、Java EE、Java ME 的概念。



1．Java SE/EE/ME 的概念

Java SE（Java Platform，Standard Edition）称为 J2SE，它允许开发和部署在桌面、服务器、嵌入式环境和实时环境中使用的 Java 应用程序。Java SE 包含了支持 Java Web 服务开发的类，并为 Java Platform，Enterprise Edition（Java EE）提供基础。

Java EE（Java Platform，Enterprise Edition）称为 J2EE，是企业版本，可以帮助开发和部署可移植、健壮、可伸缩且安全的服务器端 Java 应用程序。Java EE 是在 Java SE 的基础上构建的，它提供 Web 服务、组件模型、管理和通信 API，可以用来实现企业级的面向服务体系的结构（Service-Oriented Architecture，SOA）和 Web 2.0 应用程序。

Java ME（Java Platform，Micro Edition）称为 J2ME，它为在移动设备和嵌入式设备（如手机、PDA、电视机顶盒和打印机）上运行的应用程序提供一个健壮且灵活的环境。Java ME 包括灵活的用户界面、健壮的安全模型、许多内置的网络协议以及对可以动态下载的联网和离线应用程序的丰富支持。基于 Java ME 规范的应用程序只需编写一次，就可以用于许多设备，而且可以利用每个设备的本机功能。

2．JDK 和 JRE 概念

JDK 是 Java Develop Kit（Java API 包）的缩写，SDK 是 Software Develop Kit 的缩写，以前 JDK 称为 Java Software Develop Kit，后来出了 1.2 版本后，就改名叫 JDK。JRE 是 Java Runtime Environment 的缩写，Java 程序必须在 JRE 环境中运行。如果只需要运行 Java 程序而不是进行开发，那么安装 JRE 即可。但是如果要进行 Java 相关的开发，那么则必须安装 JDK，JDK 不仅包含 JRE（也就是说 JRE 是 JDK 的子集），还包含与 Java 开发相关的各种文件和 JDK 的 Java 源码。

3．选择合适的文件进行下载

学习 JSP 应该学习属于企业级应用的 Java EE，但是 Java SE 也提供基础，究竟学哪个，许多人都搞不清楚，其实一般用户选择 Java SE Development Kit 就可以了。Java EE 是在 Java SE 基础上构建的，一般称作 Java EE SDK（Software Development Kit），它要比 Java SE 多一些内容，在 http://java.sun.com/javaee/downloads/index.jsp 网站上可以看到 Java EE 包括了 GlassFish v3、GlassFish v3 Web Profile、GlassFish v2.1.1、Java EE 6 Samples、Java EE 5 Samples、API Documentation 和 Tutorial 中的若干个，而 GlassFish 是可以替代 Tomcat 的适用于 Java EE 的免费和开源的应用服务器。

一般来讲，说 JDK 时就是在说 Java SE 的 JDK，所以只需要下载 Java SE SDK 即可，Java EE 是企业级应用的（Java ME 应用于嵌入式设备，如手机 Java 程序开发）。

在 IE 地址栏中输入“http://Java.sun.com/”，在打开网页的导航栏中将鼠标指向 Downloads 标签，在出现的下拉列表框中选择 Java SE 选项，或者直接在地址栏中输入“http://java.sun.com/javase/downloads/index.jsp”，在打开的网页中显示了几个选项，包括 Java Platform、Standard Edition、Java SE Development Kit (JDK) Bundles、Additional Resources，单击 Download JDK 按钮，如图 7-2 所示。

打开如图 7-3 所示的页面，在其中可以选择 Platform（平台）为 Linux、Solaris 或其他 64 位操作系统平台，默认为 Windows，保持默认，单击 Download 按钮开始下载。当然也可以选择使用 Sun Download Manager，然后开始慢慢等待下载完成。

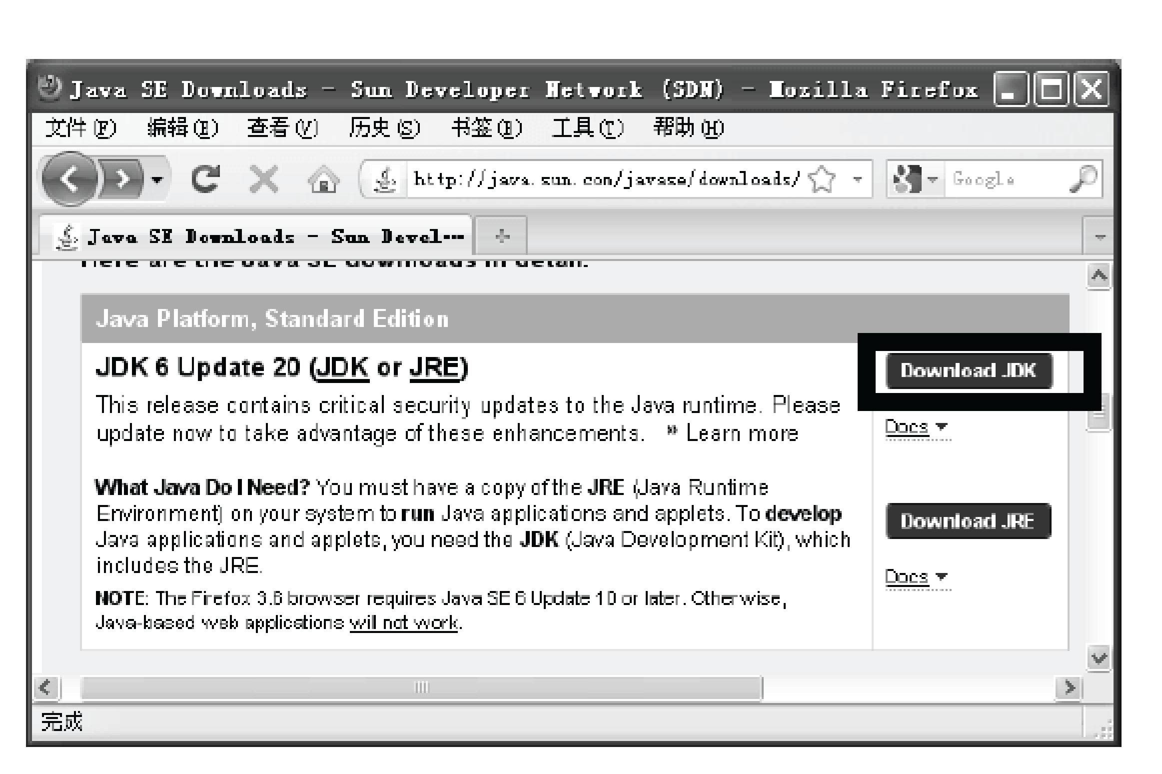


图 7-2 下载 JDK

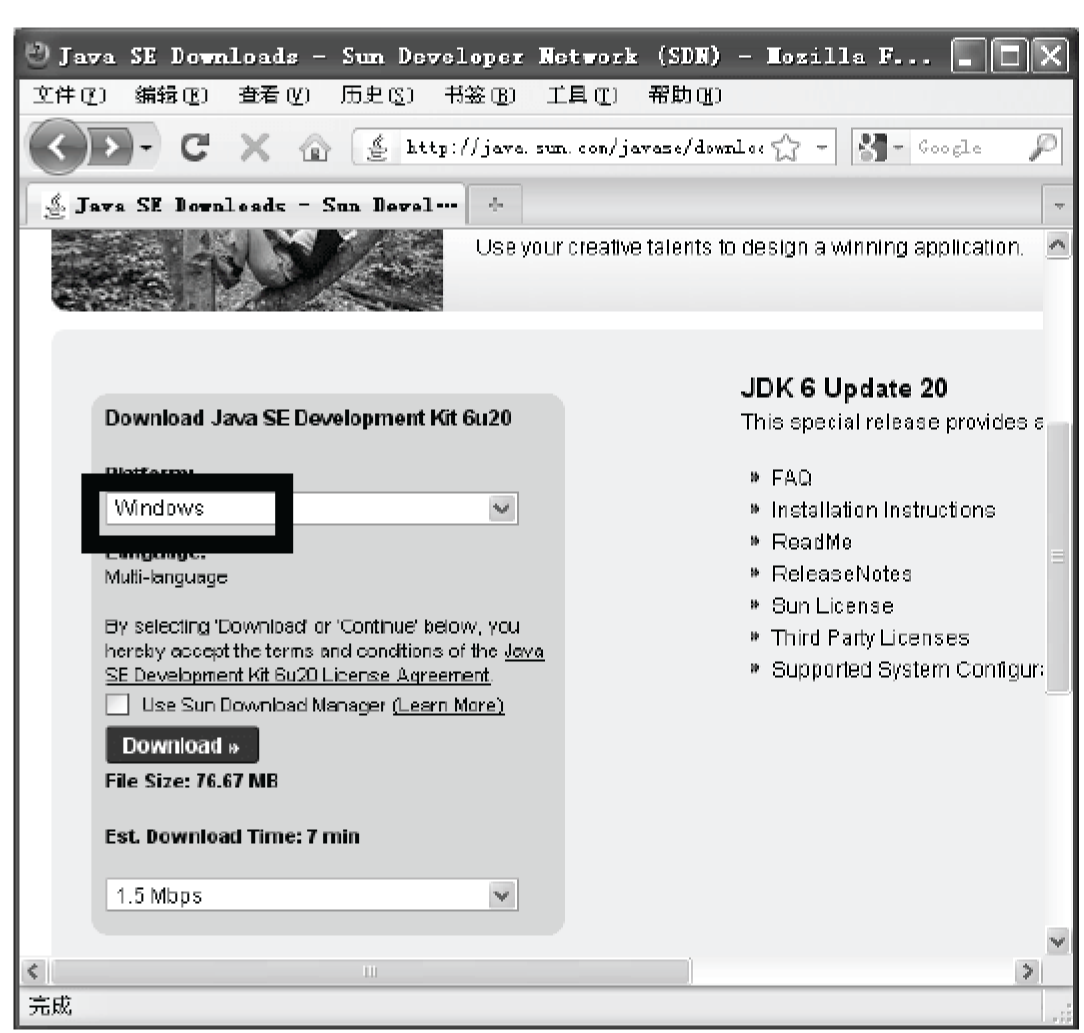


图 7-3 下载页面

下载完成后，可以发现 jdk-6u20-windows-i586.exe 的大小为 76.6MB，如图 7-4 所示。



图 7-4 jdk-6u20-windows-i586.exe 文件

7.2.2 JDK 的安装

JDK 的安装过程比较简单，双击已经下载的 jdk-6u20-windows-i586.exe，打开许可协议，单击“接受”按钮，如图 7-5 所示。

接下来进入自定义安装，把它安装到 C:\Java\jdk1.6.0_20 文件夹下（默认是安装在 C:\Program Files\Java\jdk1.6.0_20\），如图 7-6 所示。





图 7-5　单击“接受”按钮



图 7-6　自定义安装

在验证、复制新文件、注册产品等安装过程中，会弹出安装公共 JRE 的对话框，这个公共 JRE 和 JDK 没有关系，在 JDK 目录中也包含用于开发的 JRE 环境，和公共 JRE 只有少量差别，按照默认配置进行安装。安装结束后会弹出成功完成对话框，单击“完成”按钮即可，如图 7-7 所示。也可以进行注册，注册是免费的，但是初学者不必注册，因为其内容对初学者没有太多帮助。

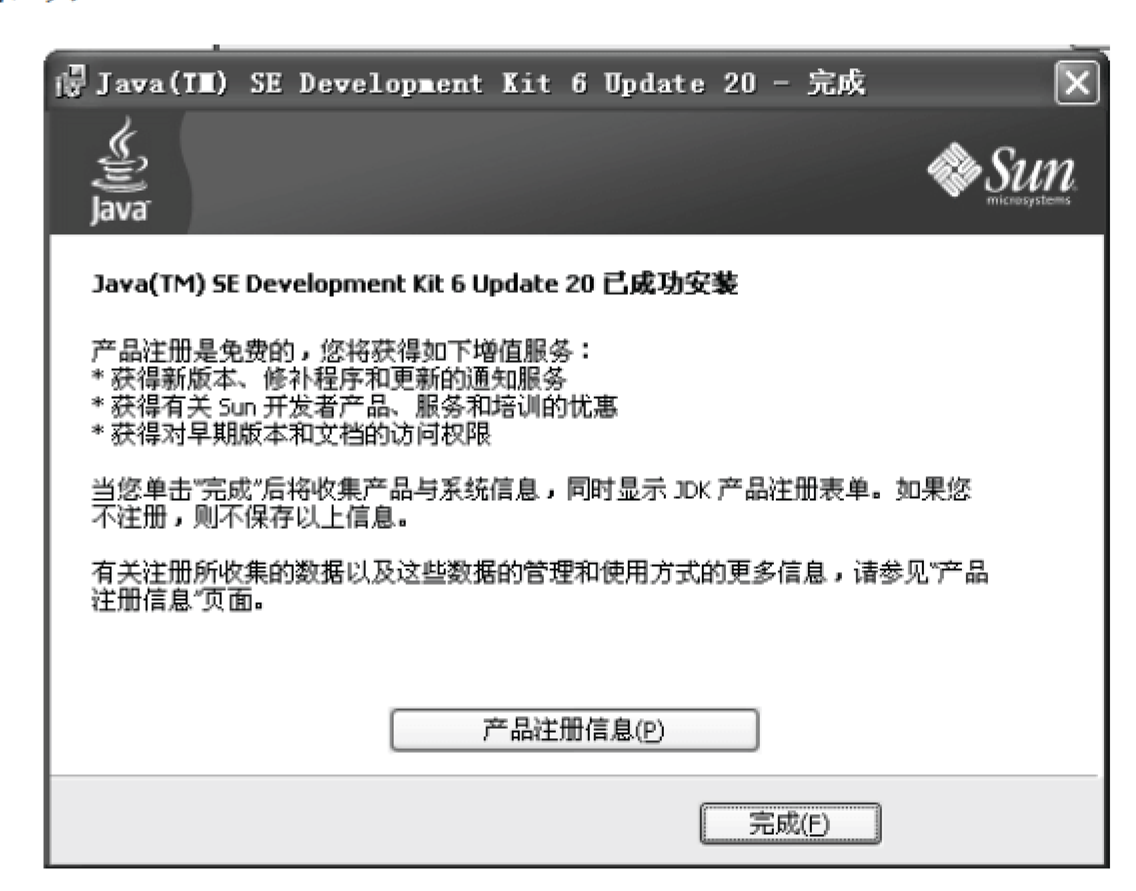


图 7-7　完成安装

7.2.3　设置 JDK 环境变量

设置了环境变量后，就可以在任意一个地方打开 cmd 命令提示符，进行 Java 文件的编译和运行，也为一些 shell 脚本的运行提供了方便，而不用每次都输入类似 C:\Java\jdk1.6.0_20\bin\java.exe 来运行 Java 字节码文件。

要设置的有 3 个环境变量：JAVA_HOME、CLASSPATH 和 Path，分别是 JavaJDK 的路径、Java class 字节码文件路径和系统路径，其中 Path 系统路径是系统中本来就有的。

右击“我的电脑”图标，在弹出的快捷菜单中选择“属性”命令，打开“系统属性”

对话框，选择“高级”选项卡，如图 7-8 所示。

单击“环境变量”按钮，弹出“环境变量”对话框，在“系统变量”栏中单击“新建”按钮，弹出“新建系统变量”对话框，在“变量名”文本框中输入“JAVA_HOME”，在“变量值”文本框中输入 JDK 的路径“C:\Java\jdk1.6.0_20”，如图 7-9 所示。

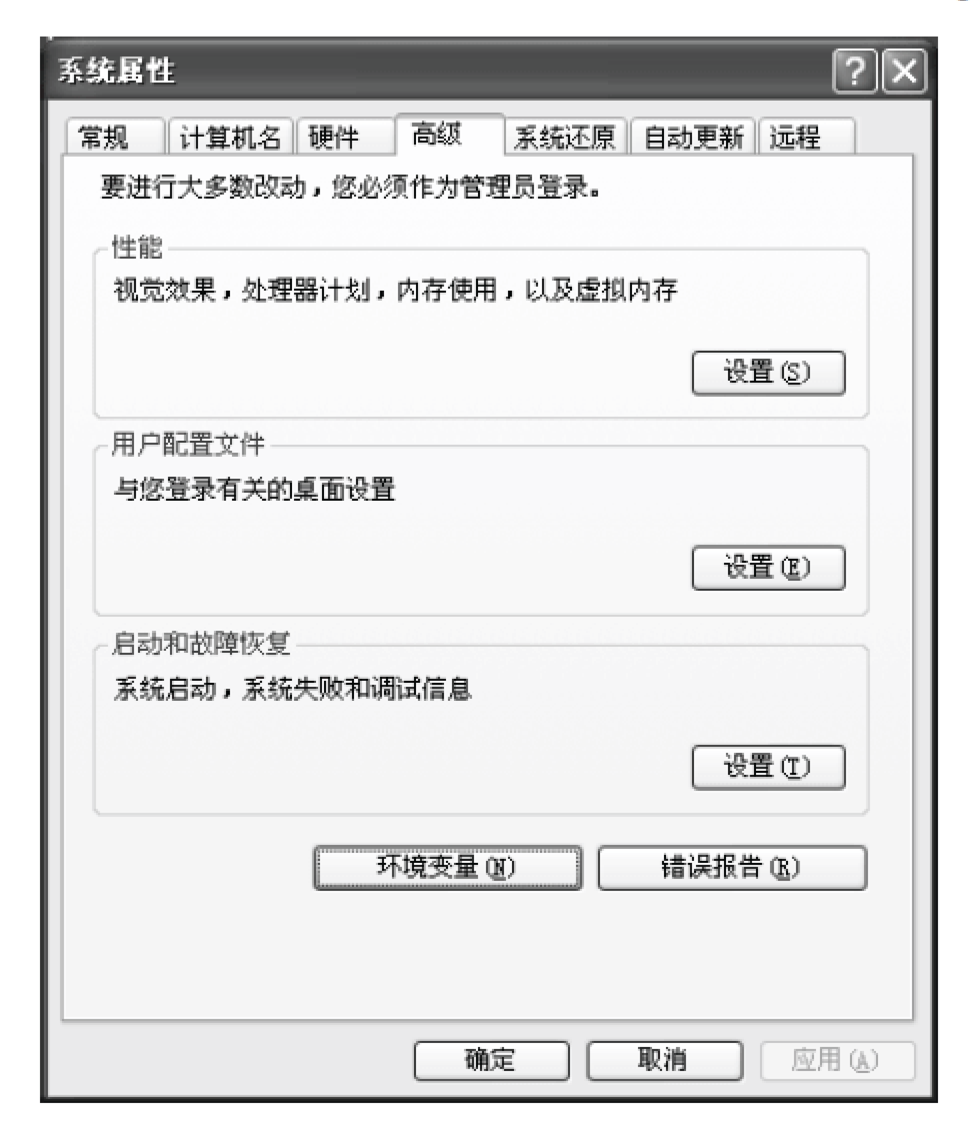


图 7-8 “高级”选项卡

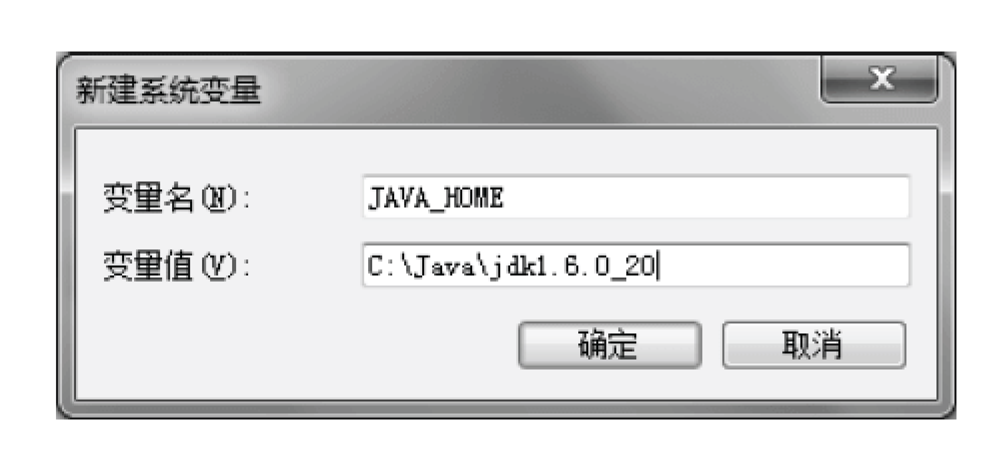


图 7-9 设置环境变量 JAVA_HOME

单击“确定”按钮，JAVA_HOME 变量设置完毕。用同样的方法，设置 CLASSPATH 变量，在“变量值”文本框中输入“.;”(此时点和分号必须是英文标点)，单击“确定”按钮。CLASSPATH 的变量值表示字节码的路径是在当前路径下，如图 7-10 所示。

Path 环境变量是系统本身就有的，在“系统变量”栏中找到 Path 变量，单击它，并单击“编辑”按钮，在打开的“编辑系统变量”对话框中的“变量值”文本框中按 Home 键，表示将鼠标光标移动到该文字区域的开始位置，输入“%JAVA_HOME%\bin;”，表示将刚才设置的 JAVA_HOME 变量下的 bin 文件夹（包括 Java 的各种 exe 可执行文件）的路径加入 Path 系统变量，如图 7-11 所示，这样系统就可以自动搜索到 Java 程序的路径。

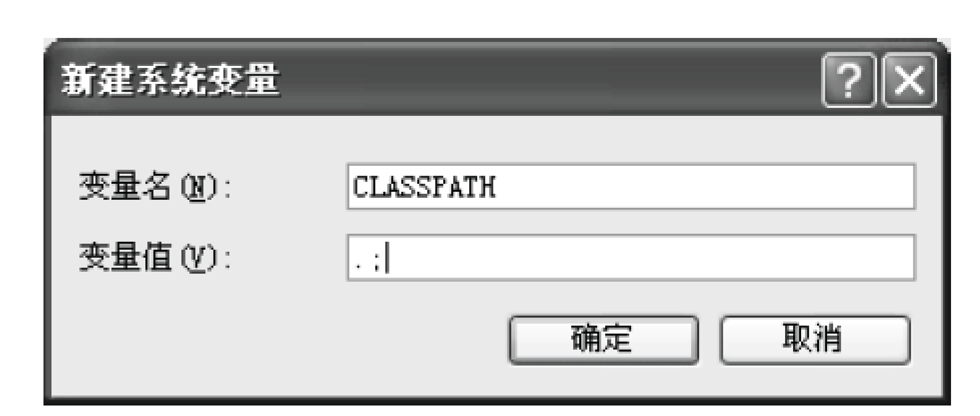


图 7-10 设置环境变量 CLASSPATH



图 7-11 Path 环境变量设置

7.2.4 测试 JDK 环境变量

选择“开始”→“运行”命令，在弹出的“运行”对话框中输入“notepad”，打开记事本，在其中输入以下内容：



```
public class hello{
        public static void main(String args[]){
              System.out.println("Hello,Java");
        }
        }
```

按 Ctrl+S 组合键，弹出“另保存”对话框，输入文件名“C:\Java\hello.java”，单击“保存”按钮。下面打开 cmd 命令提示符窗口，使用 Java 编译器来编译此 hello.java 文件。选择“开始”→“运行”命令，在打开的对话框中输入“cmd”，打开了 cmd 命令提示符窗口，输入以下命令：

```
cd \
cd java
javac hello.java
java hello
```

如果出现了“Hello,Java”信息，则证明 JDK 环境变量设置成功，如图 7-12 所示。

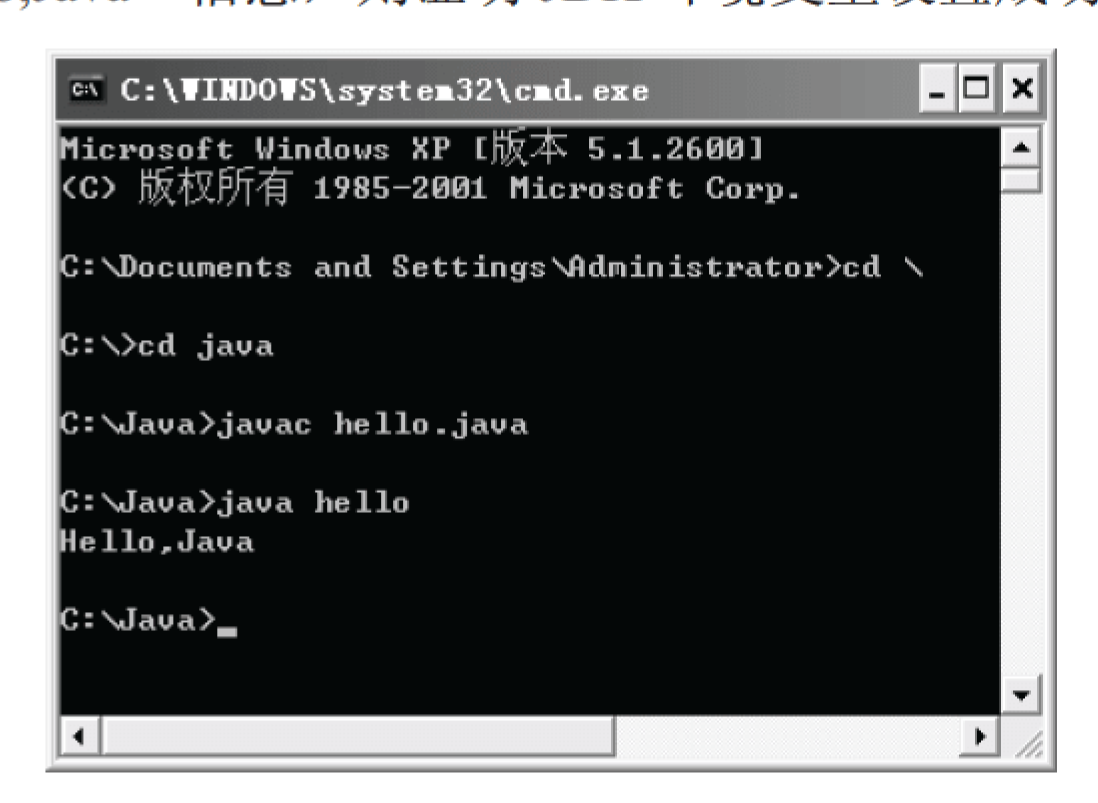


图 7-12　测试 JDK 环境变量

7.3　安装与配置 Tomcat

7.3.1　安装 Tomcat

下面进行 Tomcat 服务器的安装和配置，Tomcat 可以从 http://tomcat.apache.org/download-60.cgi 网站上进行下载，可以选择 32-bit Windows zip，也可以选择 32-bit/64-bit Windows Service Installer 的二进制安装包。

如图 7-13 所示，apache-tomcat-6.0.26.exe 和 apache-tomcat-6.0.26-windows-x86.zip 的文件大小分别为 6.61MB 和 6.65MB，推荐下载.exe 格式的安装包。

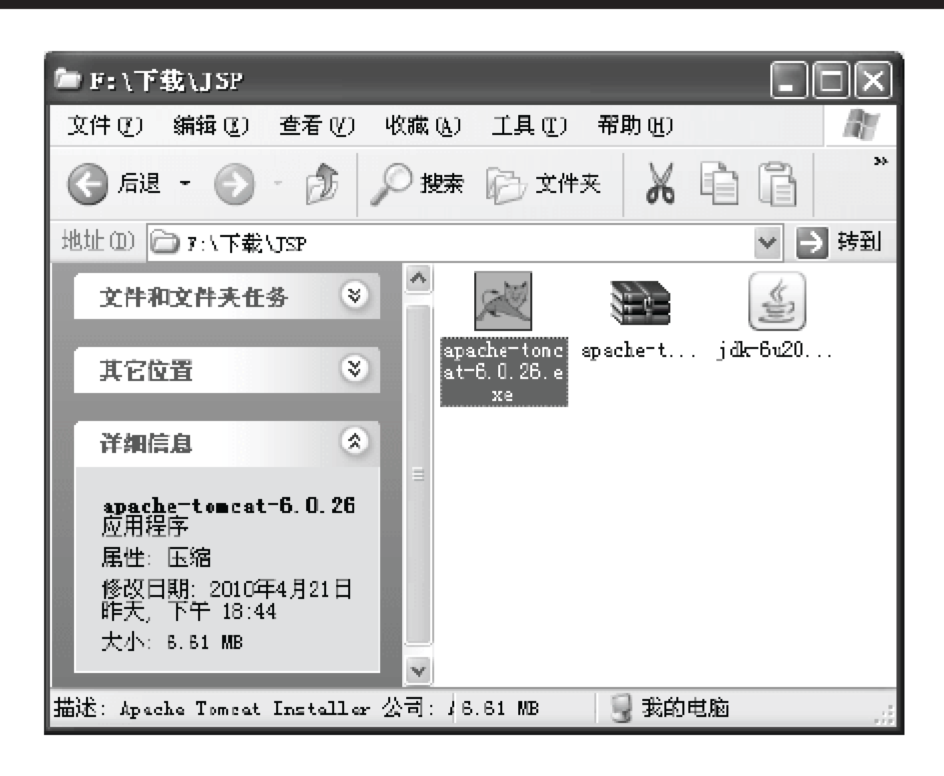


图 7-13 apache-tomcat-6.0.26.exe 和 apache-tomcat-6.0.26-windows-x86.zip 文件

Tomcat 的安装也非常简单，如果下载的是 zip 文件，将 apache-tomcat-6.0.26-windows-x86.zip 解压到 C:\tomcat-6.0.26 文件夹内，其实可以将它解压到任意一个目录，因为它是绿色版的，但是为了方便查看，把它解压到 C 盘。

如果下载的是 apache-tomcat-6.0.26.exe 文件，则直接双击进行安装。在选择安装路径时，将路径设置为 C:\Tomcat 6.0，然后进入 Tomcat 设置，此时需要设置 Tomcat 的默认端口号，默认为 8080，可以改为 80；然后设置 Tomcat 管理员登录可选项，设置用户名为 root，密码为 123456，也可以设置为其他用户名和更高复杂度的密码，如图 7-14 所示。

单击 Next 按钮，选择 Java 虚拟机所在位置，也就是 JRE 所在位置，可以选择 JDK 路径下的 JRE，即 C:\Java\jdk1.6.0_20\jre，也可以选择公共 JRE，即 C:\Program Files\Java\jre6。这里选择 JDK 路径下的 JRE，如图 7-15 所示。

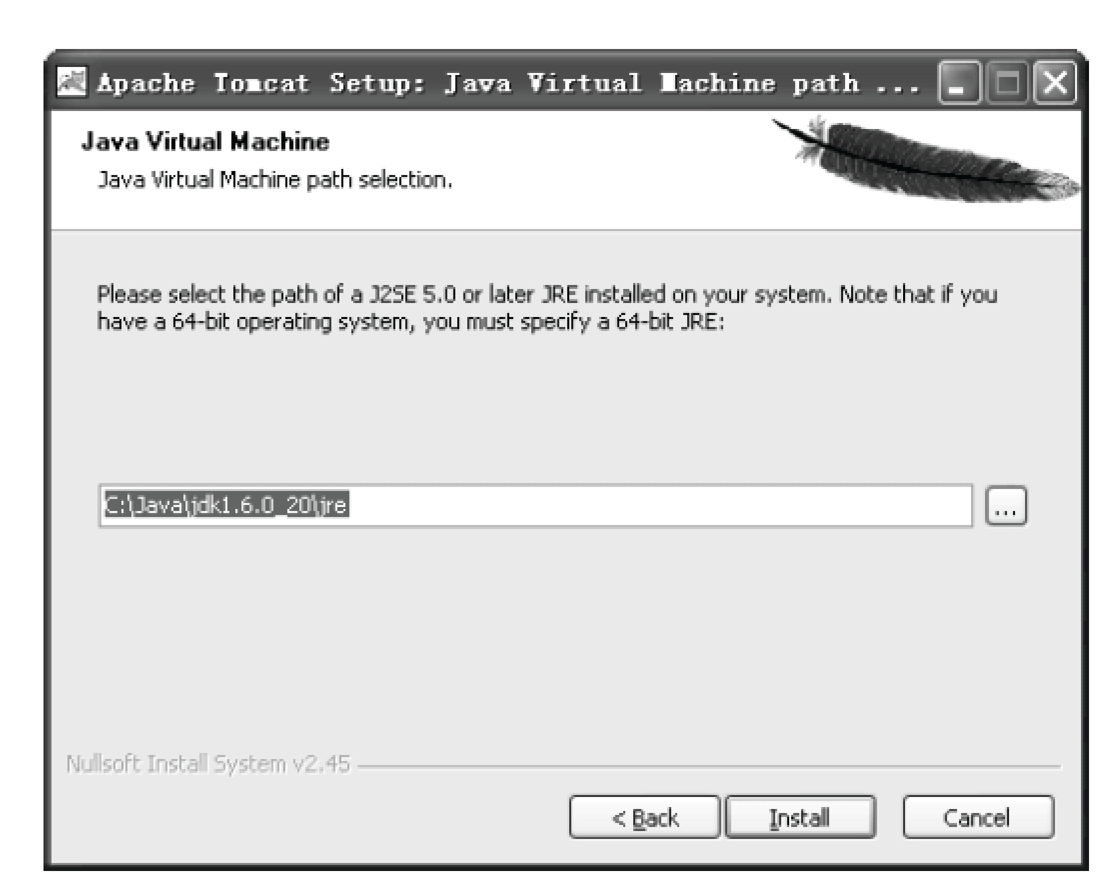



图 7-14 Tomcat 设置

图 7-15 选择 Java 虚拟机所在位置

单击 Install 按钮开始安装，片刻后即可完成。

7.3.2 测试 Tomcat

安装 Tomcat 后，任务栏中会出现 Apache Tomcat 6 的图标。绿色的三角符号代表 Tomcat 是运行状态，如果没有运行，则图标中会出现红色的方块。在运行状态下，可以



通过浏览器来访问 Tomcat 服务器所提供的 Web 服务。

打开浏览器（本例中的是火狐浏览器），在浏览器地址栏中输入“http://localhost/index.jsp”，按 Enter 键后即可看到如图 7-16 所示的网页。如果指定的 Tomcat 端口是 8080，则需要在地址栏中输入“http://localhost:8080/index.jsp”。

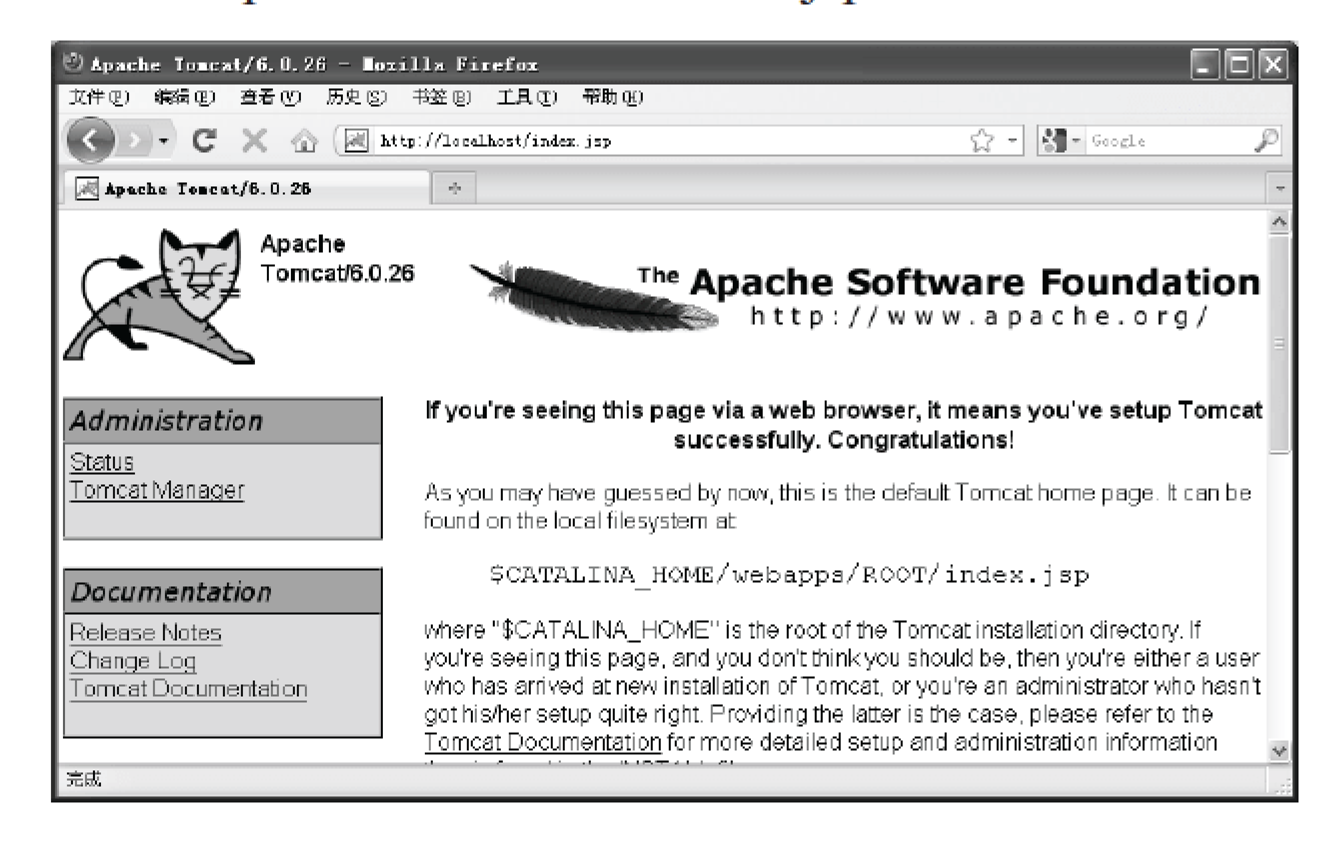


图 7-16　测试 Tomcat

如果页面出现类似<%@ page session="false" %>的语句，说明 Tomcat 无法解析 JSP，需要对 Tomcat 的 Java 部分进行配置。

7.3.3　配置 Tomcat

由于是先安装 JDK，后安装 Tomcat，所以 Tomcat 会检测到 JDK 和 JRE 自动进行配置，否则，就需要对 Tomcat 进行 Java 的配置。

选择“开始”→“程序”→Apache Tomcat 6.0→Configure Tomcat 命令，打开 Tomcat 的配置对话框；如果已经运行了 Tomcat Moniter，则右击任务栏右下角的 Tomcat Moniter 图标，在弹出的快捷菜单中选择 Configure 命令，也会弹出 Tomcat 的配置对话框。选择 Java 选项卡，在 Java Virtual Machine 文本框中输入“C:\Java\jdk1.6.0_20\jre\bin\server\jvm.dll”，如图 7-17 所示。

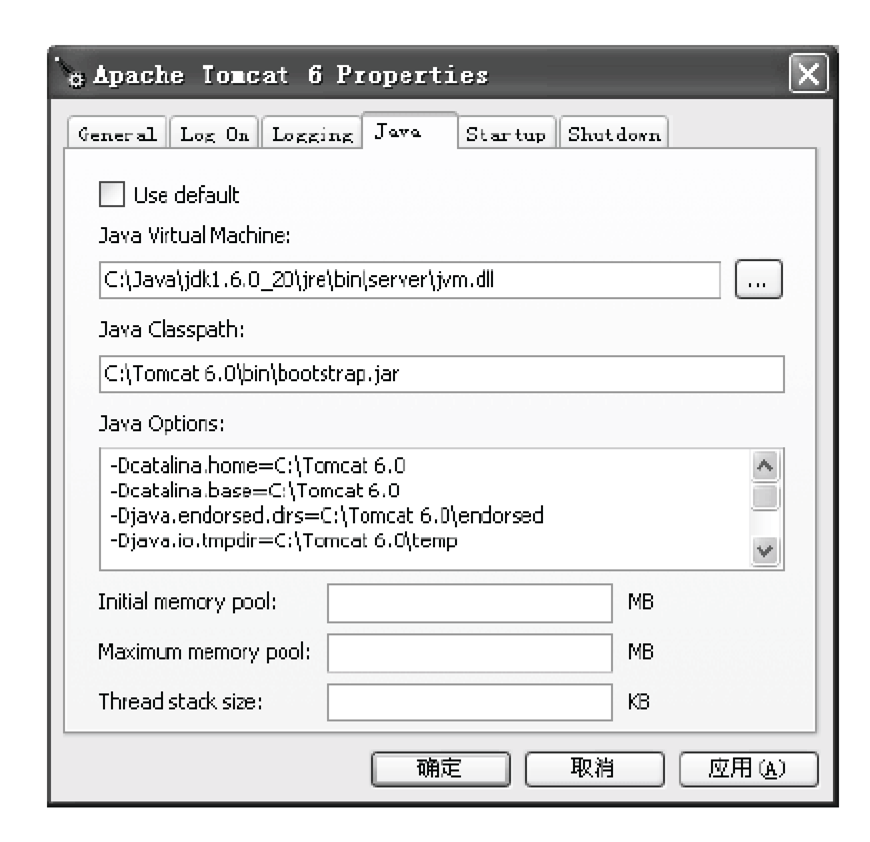


图 7-17　配置 Tomcat

这里的设置只能配置基本的服务，可使 Tomcat 正常运行，更多的配置还是要通过直接编辑 C:\Tomcat 6.0\conf\文件夹下的一些配置文件来实现。下面介绍一些 Tomcat 常用的设定。

1. Tomcat 中虚拟目录的设置

Tomcat 的默认主目录是 C:\Tomcat 6.0\webapps\ROOT，其下可以放一个或多个网页。如果想要在一个新的目录，如 C:\myweb 目录下放一个网站，并且使用 Tomcat 来浏览，那么就需要增加一个 Web Site，而不需要动原有的网站。用记事本打开 C:\Tomcat 6.0\conf\server.xml，找到以下代码：

```
<Host name="localhost"   appBase="webapps"
unpackWARs="true" autoDeploy="true"
xmlValidation="false" xmlNamespaceAware="false">
```

在下面新的一行中写入以下代码（只要是在<host>和</host>之间均可以）：

```
<Context path="/site" docBase="c:\myweb"   reloadable="true"   debug="0">
 </Context>
```

其中的 path 是相对于网站 URL 而言；/site 就是指 http://localhost/site；docBase 指的是文件在文件系统中的位置，本例中是“C:\myweb”。将网页移动到 C:\myweb\目录下，重新启动 Tomcat 后，就可以使用 http://localhost/site 网址浏览新的网页。

如果需要更改“/”目录的文件位置，只需要将上面 host 标签中的 appBase 中的值更改即可，如更改为“appBase="c:\myweb"”，这样使用 http://localhost 即可访问位于 C 盘 myweb 文件夹内的网页。

2. 修改端口号

用记事本打开 C:\Tomcat 6.0\conf\server.xml，找到以下代码：

```
<Connector port="80" protocol="HTTP/1.1"
 connectionTimeout="20000"
redirectPort="8443" />
```

将其中的 port 后面引号内的值修改为需要的端口号，如修改为“port="8080"”，重新启动 Tomcat 服务器，打开浏览器在地址栏中输入“http://localhost:8080”即可。

3. 修改首页默认值

用记事本打开 C:\Tomcat 6.0\conf\web.xml，找到以下代码（搜索 welcome-file-list 即可快速找到）：

```
<welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
```



根据自己的需要添加或修改<welcome-file>和</welcome-file-list>标签之间的内容，重启 Tomcat 后生效。

7.4　Java 开发工具——Eclipse 简介

Java 的开发工具很多，有 Eclipse、Netbeans、JBuilder、Jdeveloper、IDEA、Jcreator 和 Workshop 等，其中 Eclipse 是使用比较多的 Java 开发集成工具，其具有即时编译和运行便捷等特性，可快速构建集成 Web 应用程序。由于它是基于 Java 的开源项目，因此可以在任何平台（Java 是跨平台的）上进行运行、操作。Eclipse 的核心体系是一个动态加载插件的框架结构，因此可以使用不同的插件来实现不同的工作任务，同时也可以开发不同的插件来不断完善 Eclipse 的功能。

7.4.1　下载和安装 Eclipse

首先到 Eclipse 官方网站 http://www.eclipse.org/downloads/下载 Eclipse 软件包，对于开发 Web 应用程序的用户建议下载 Eclipse IDE for Java EE Developers（190MB），文件名是 eclipse-jee-galileo-SR2-win32.zip，其中 SR2 是开发版本，可能会有所不同，如图 7-18 所示。



图 7-18　下载和安装 Eclipse

Eclipse 的安装也非常简单，因为 Eclipse 是绿色软件，只需要将压缩包解压到一个目录即可，如把它解压到 C:\eclipse。

7.4.2　运行和配置 Eclipse

运行 C:\eclipse 目录下的 eclipse.exe，弹出 Workspace Launcher 对话框，为了方便，设置 Workspace 的位置在 C:\eclipse\workspace。如图 7-19 所示是默认状态下的 Workspace Launcher 对话框。Eclipse 的 Welcome 页面如图 7-20 所示。

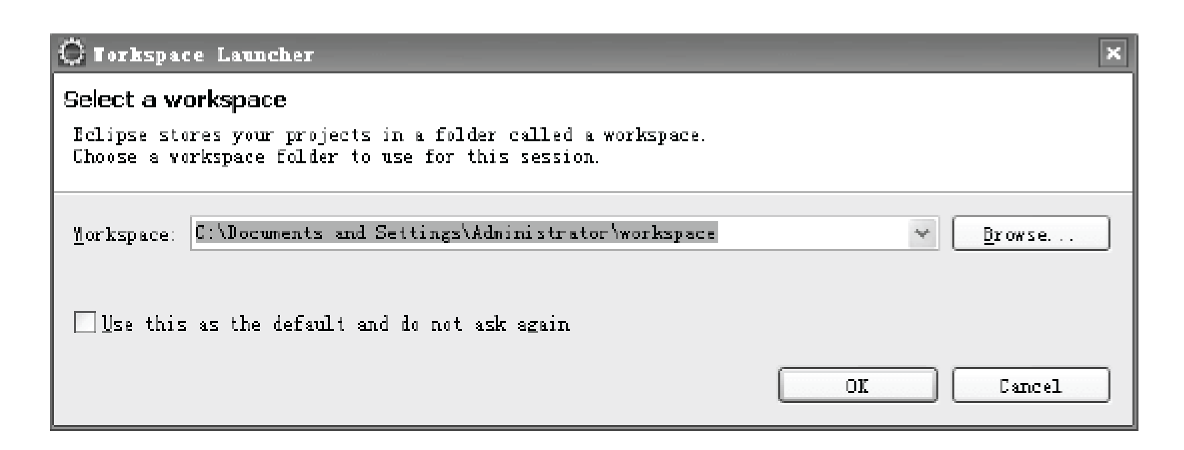


图 7-19　默认状态下的 Workspace Launcher 对话框



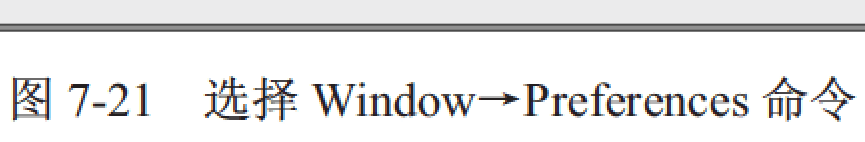
图 7-20　Eclipse 的 Welcome 页面

下面配置 Eclipse 的 Tomcat 运行环境，这样就可以方便地进行 Tomcat 操作及应用部署等工作。

（1）在主界面上选择 Window→Preferences 命令，如图 7-21 所示。

（2）在打开的 Preferences 窗口中，选择左侧 Server 下的 Runtime Environments 选项，如图 7-22 所示。




图 7-21　选择 Window→Preferences 命令

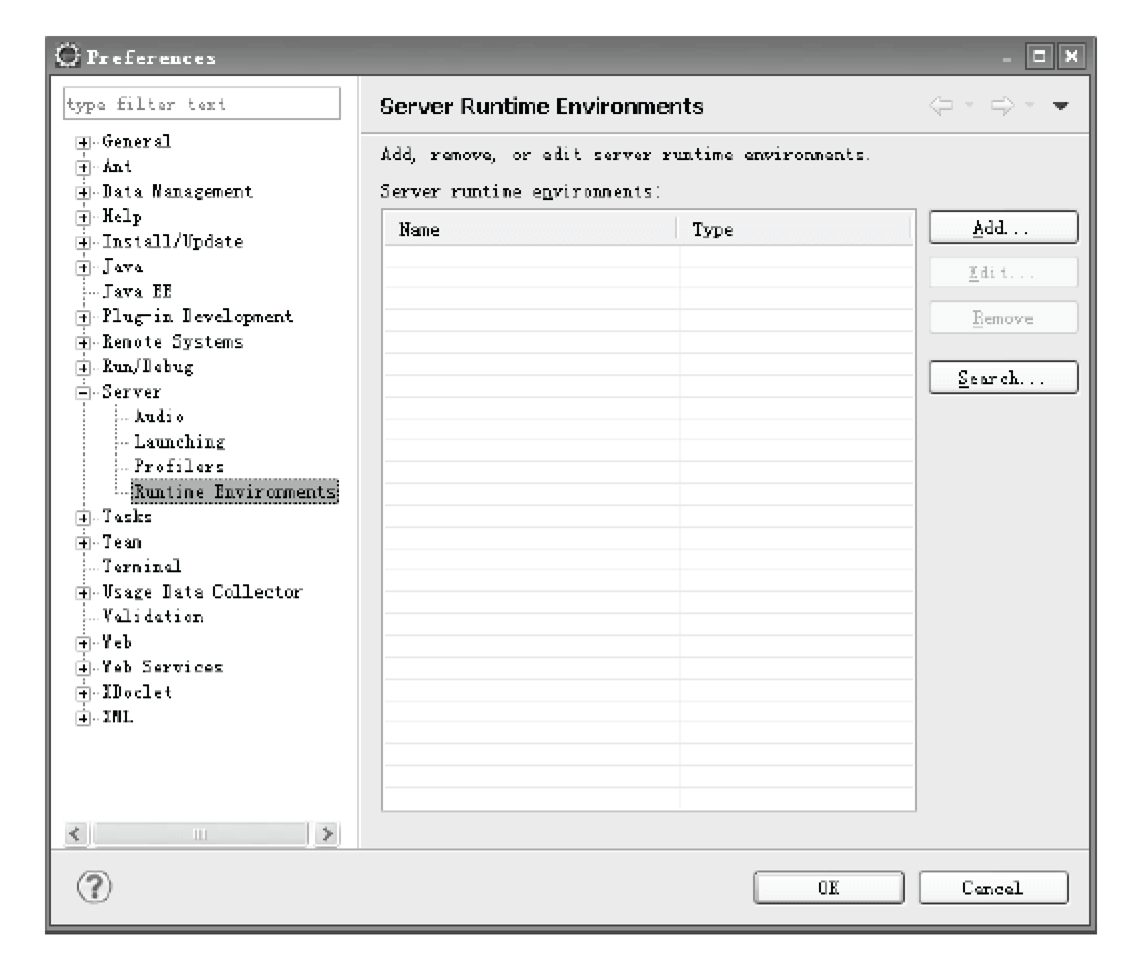


图 7-22　选择 Runtime Environments 选项

（3）在右侧单击 Add 按钮，在打开的窗口中的列表框中选择 Apache 下的 Apache Tomcat v6.0 选项，这里要和 Tomcat 版本一致，如图 7-23 所示。



（4）单击 Next 按钮，单击 Tomcat installation directory 文本框后的 Browser 按钮，设置 Tomcat 6.0 的路径为 C:\Tomcat 6.0，如图 7-24 所示。

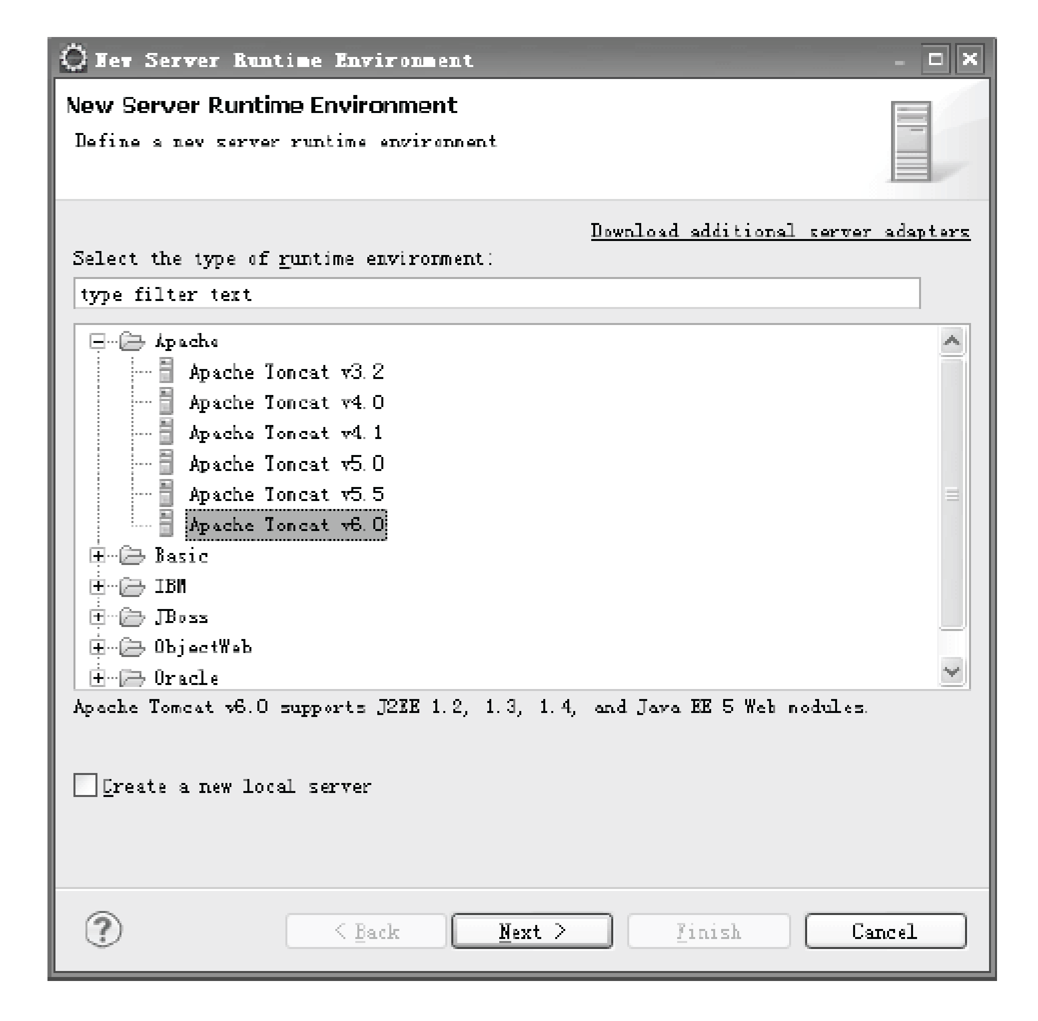


图 7-23　选择 Apache Tomcat v6.0 选项

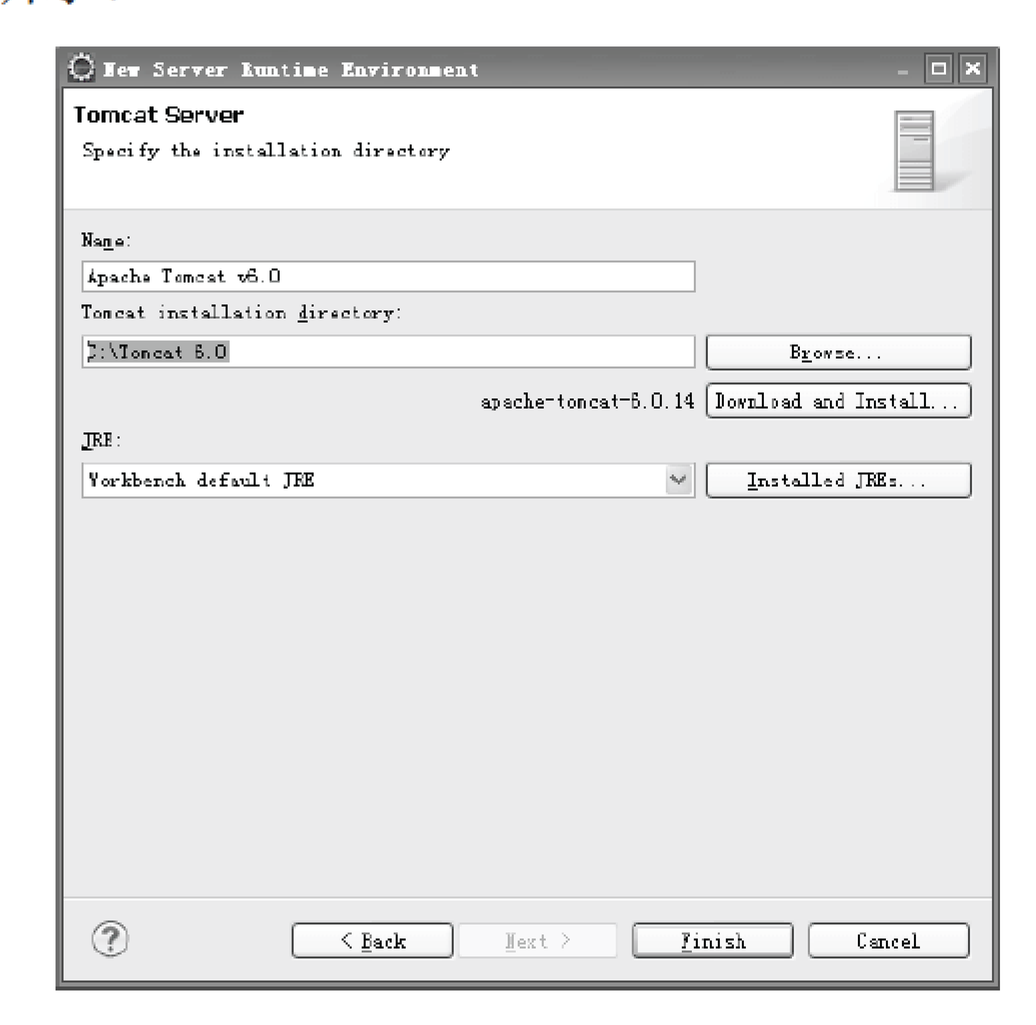


图 7-24　设置 Tomcat 6.0 路径

（5）单击 Installed JREs 按钮，在打开的对话框中显示已经自动配置好了 JDK，如果没有，则需要单击 Add 按钮添加一个新的标准 JRE，如图 7-25 所示。

（6）单击 OK 按钮，返回 New Server Runtime Environment 窗口，单击 Finish 按钮，返回 Preferences 窗口，单击 OK 按钮，Eclipse 的 Tomcat 服务器运行环境即配置完成，如图 7-26 所示。

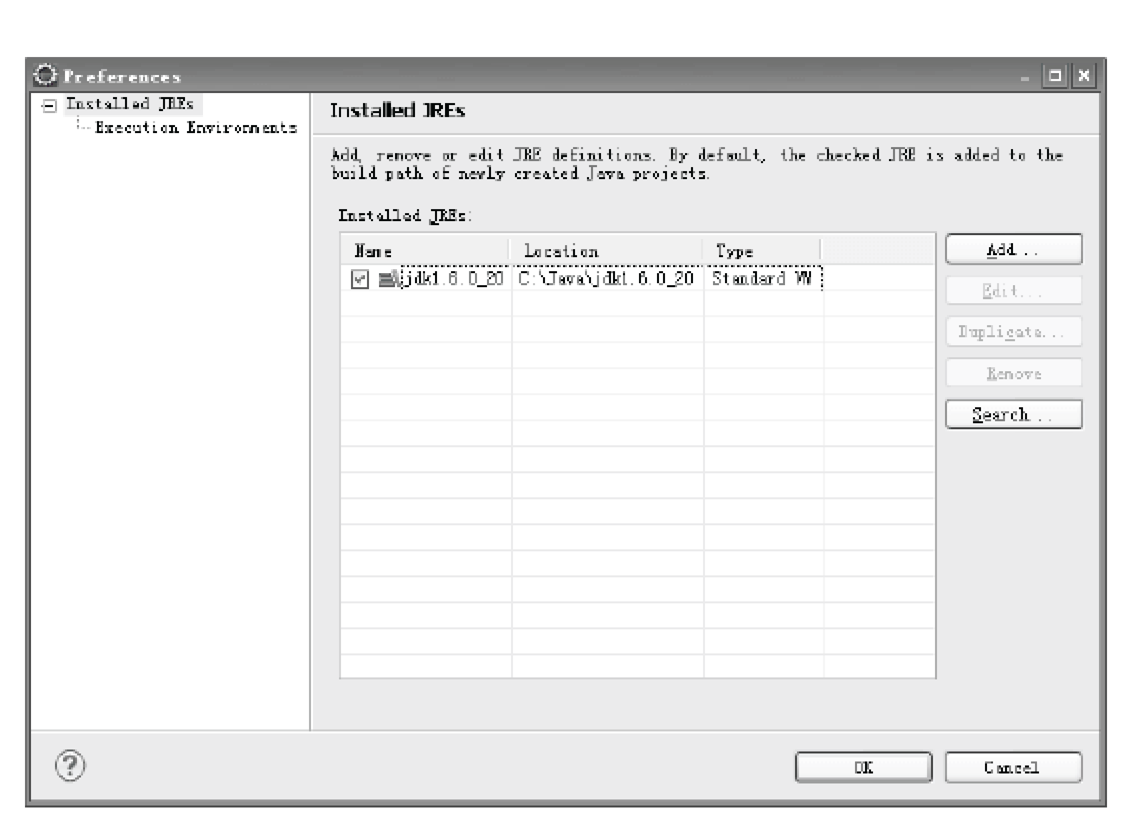


图 7-25　添加一个新的标准 JRE

图 7-26　完成 Eclipse 的 Tomcat 服务器运行环境配置

7.4.3　使用 Eclipse 开发一个简单的 Web 应用程序

1．新建一个工程及网页

（1）首先新建一个工程，在主界面中选择 File→New→Project 命令，如图 7-27 所示。

（2）弹出 New Project 对话框，在中间的列表框中选择 Web 下的 Dynamic Web Project 选项，即动态网页选项，然后单击 Next 按钮，如图 7-28 所示。

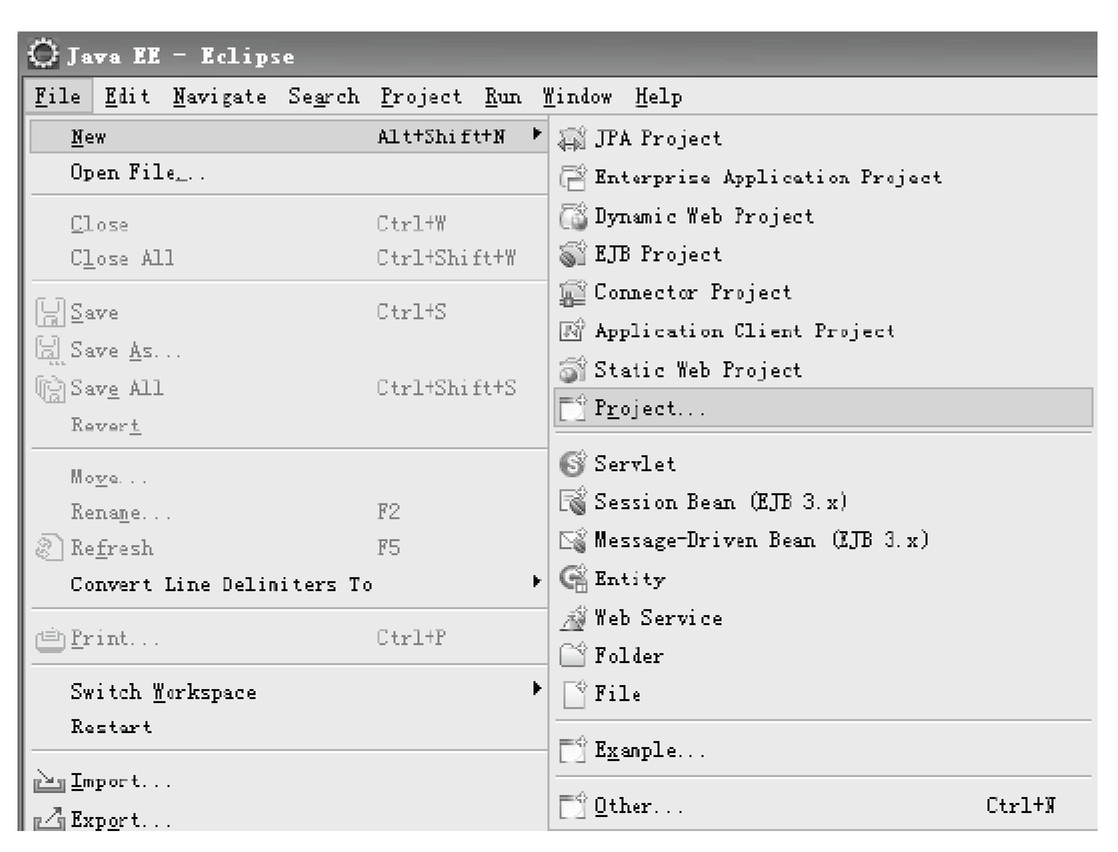



图 7-27 选择 File→New→Project 命令

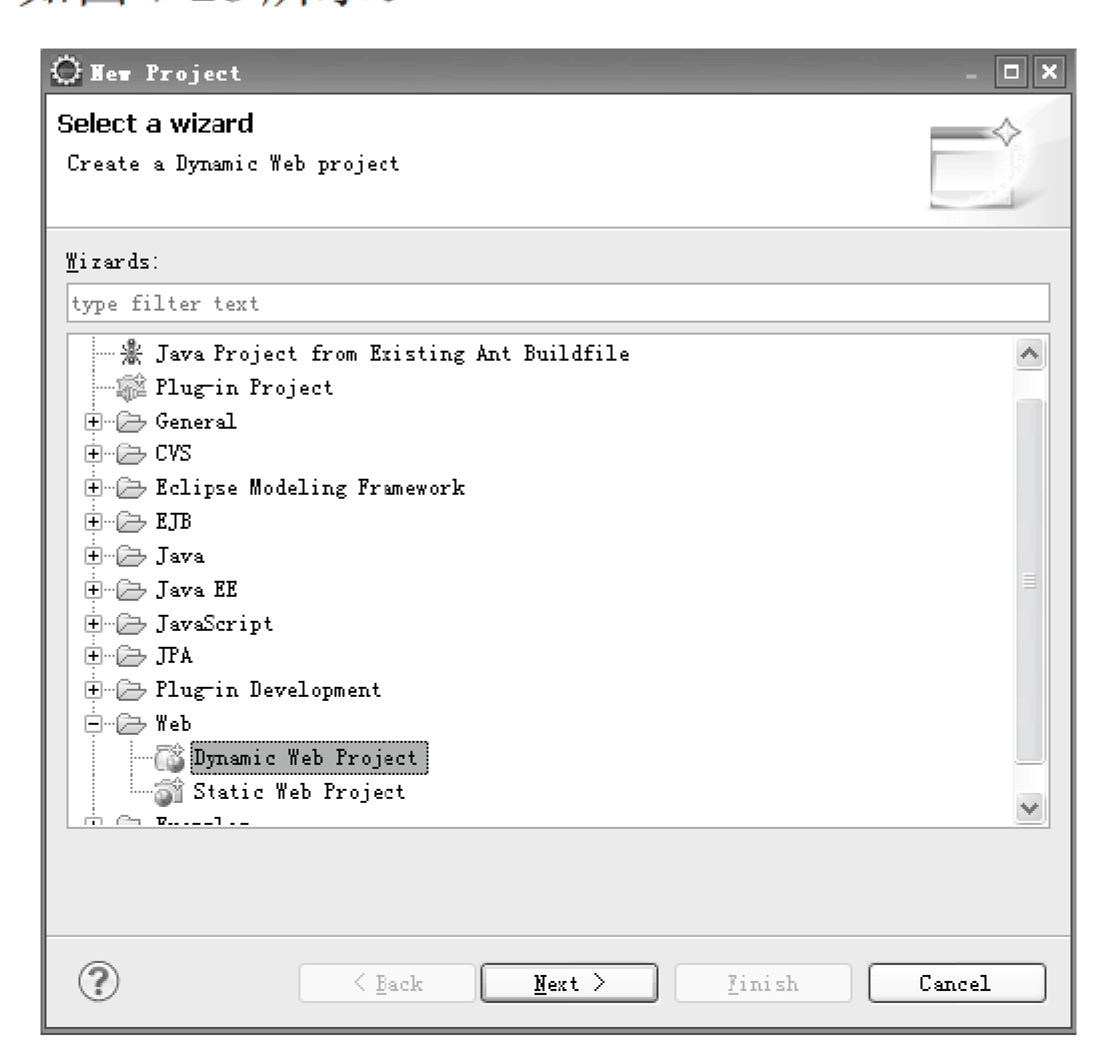


图 7-28 New Project 对话框

（3）弹出 New Dynamic Web Project 对话框，在 Project name 文本框中输入项目的名称，如“HelloWorld”，由于已经配置好 Tomcat 服务器运行环境，其他选项保持默认即可，单击 Next 按钮，如图 7-29 所示。

（4）接下来是更多细节方面的配置，这里是为 Java 程序所设置的配置，暂时不需要使用 Java，单击 Next 按钮（也可以单击 Finish 按钮结束），如图 7-30 所示。



图 7-29 New Dynamic Web Project 对话框

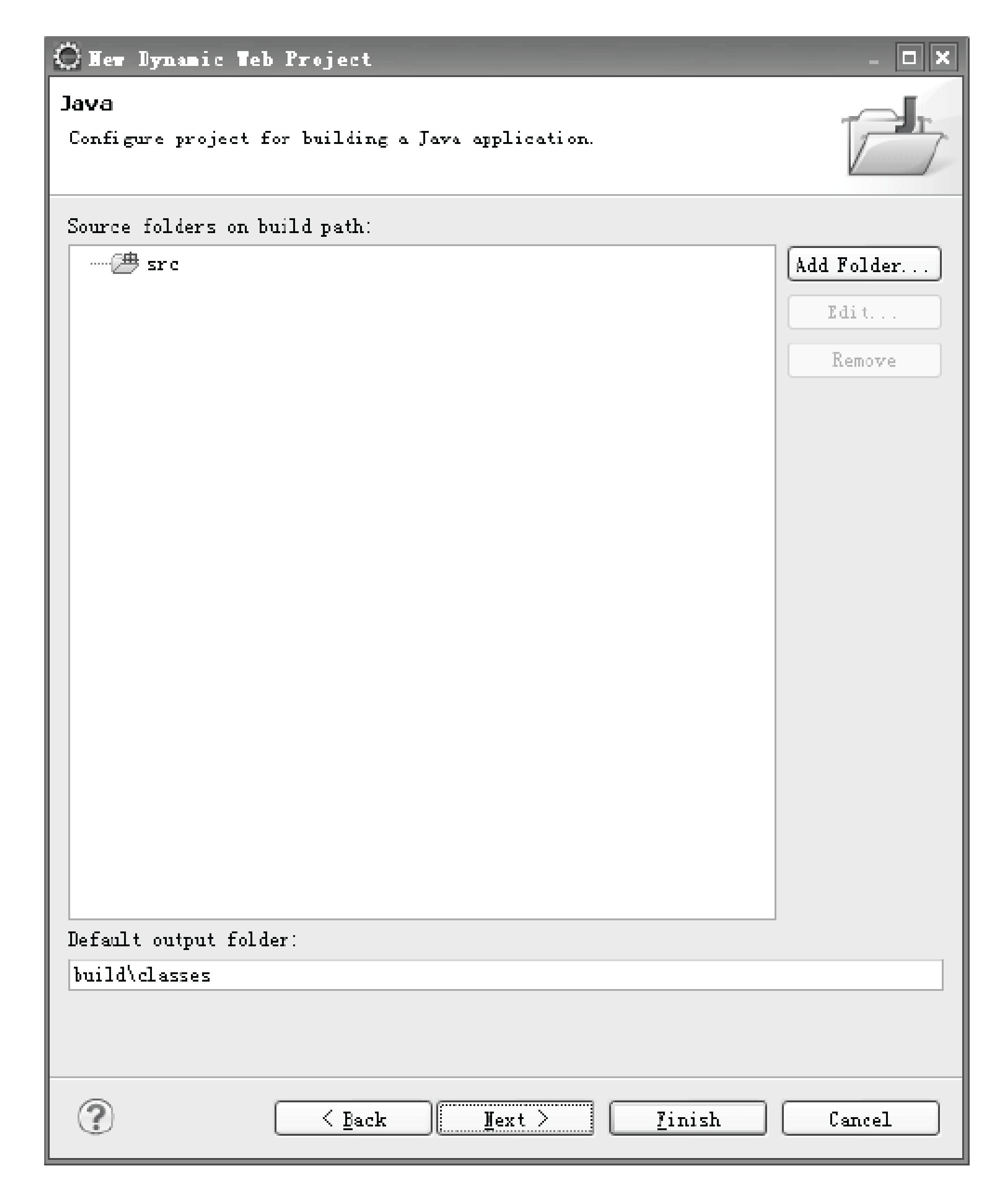


图 7-30 暂不设置 Java

（5）设置网页模块的名称，保持默认即可，如图 7-31 所示，单击 Finish 按钮。



（6）添加一个 jsp 网页文件。在 Project Explorer 项目浏览窗口中，在 HelloWorld 项目上单击鼠标右键，在弹出的快捷菜单中选择 New→JSP 命令，如图 7-32 所示。

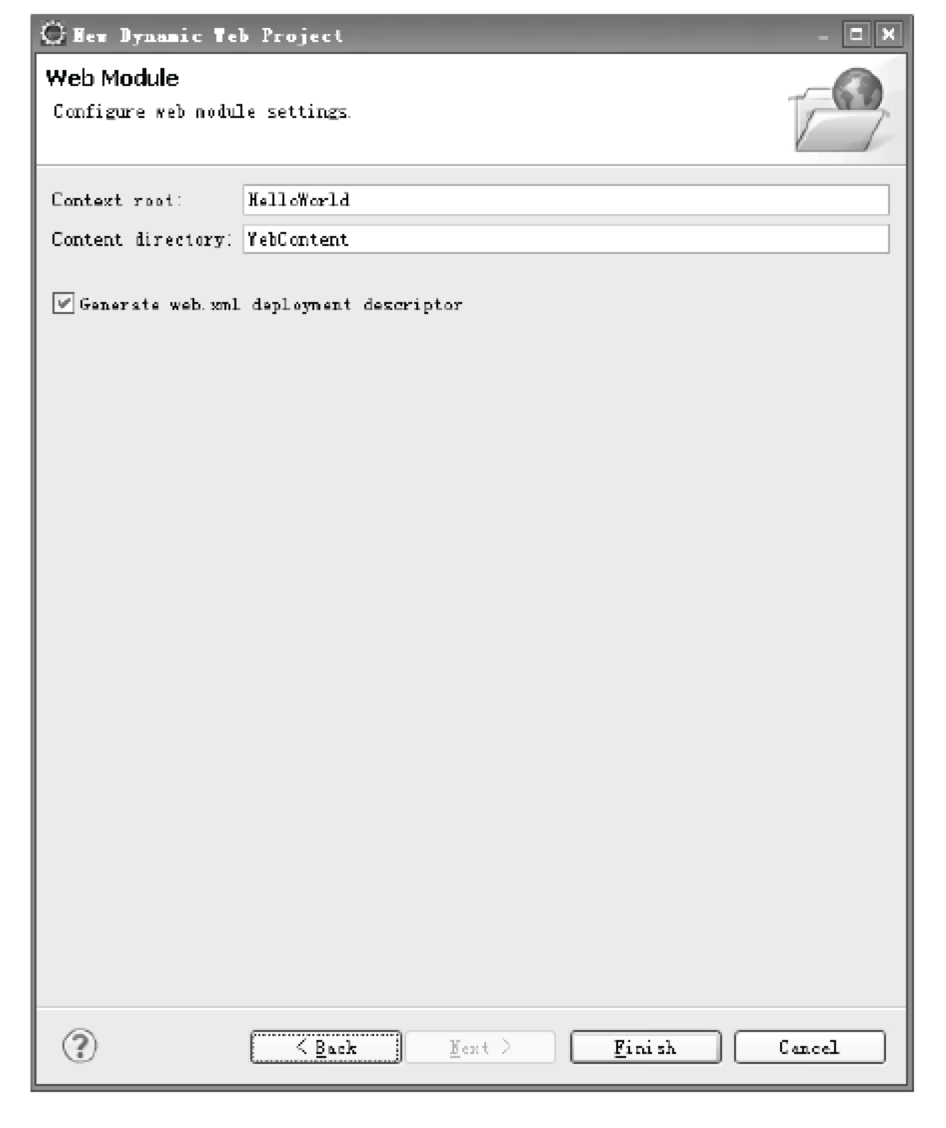


图 7-31　设置网页模块的名称

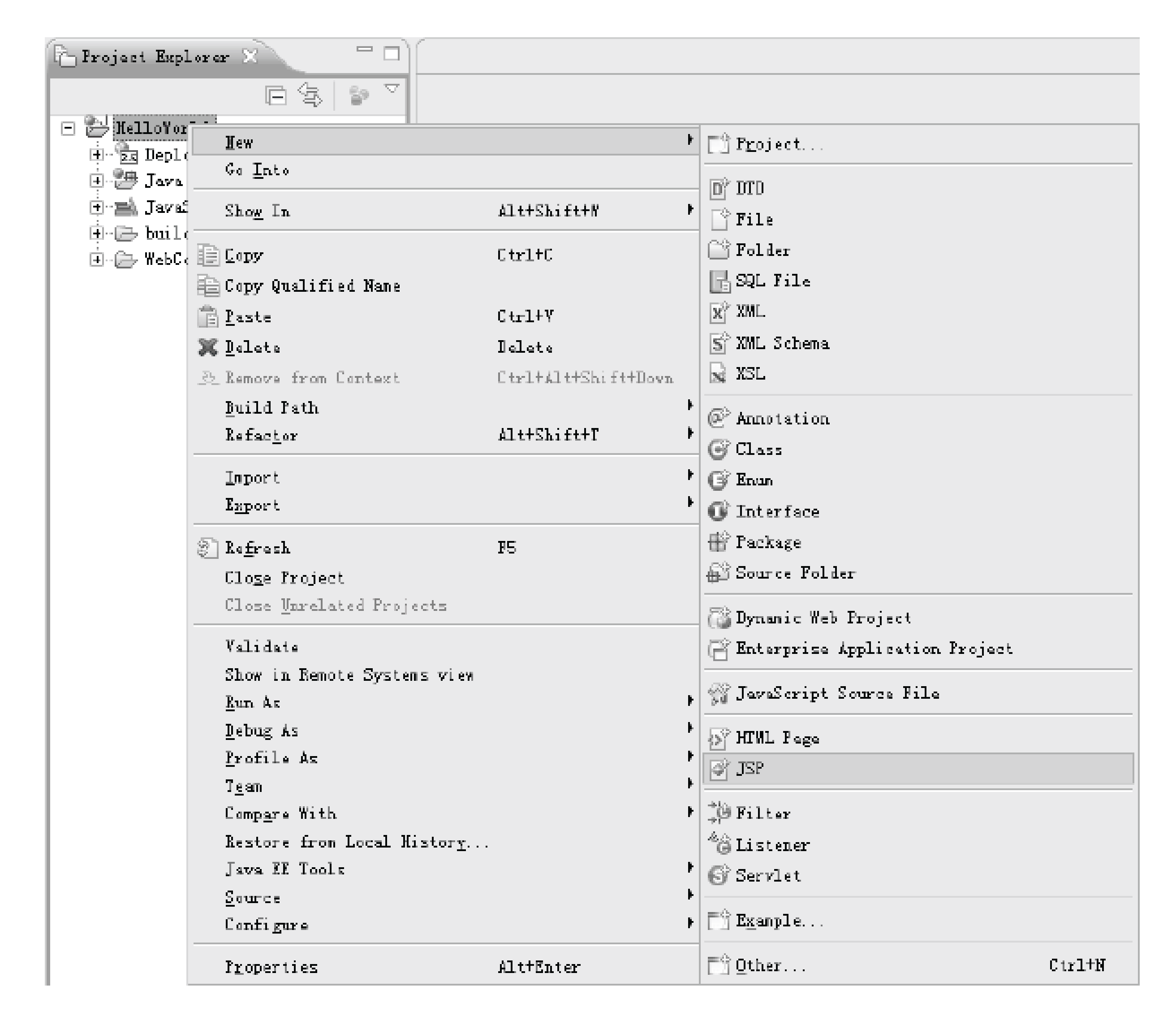


图 7-32　Project Explorer 项目浏览窗口

（7）在 New JavaServer Page 窗口的 File name 文本框中输入“index.jsp”，如图 7-33 所示。

（8）选择 JSP 网页模板，保持默认即可，如图 7-34 所示，单击 Finish 按钮。

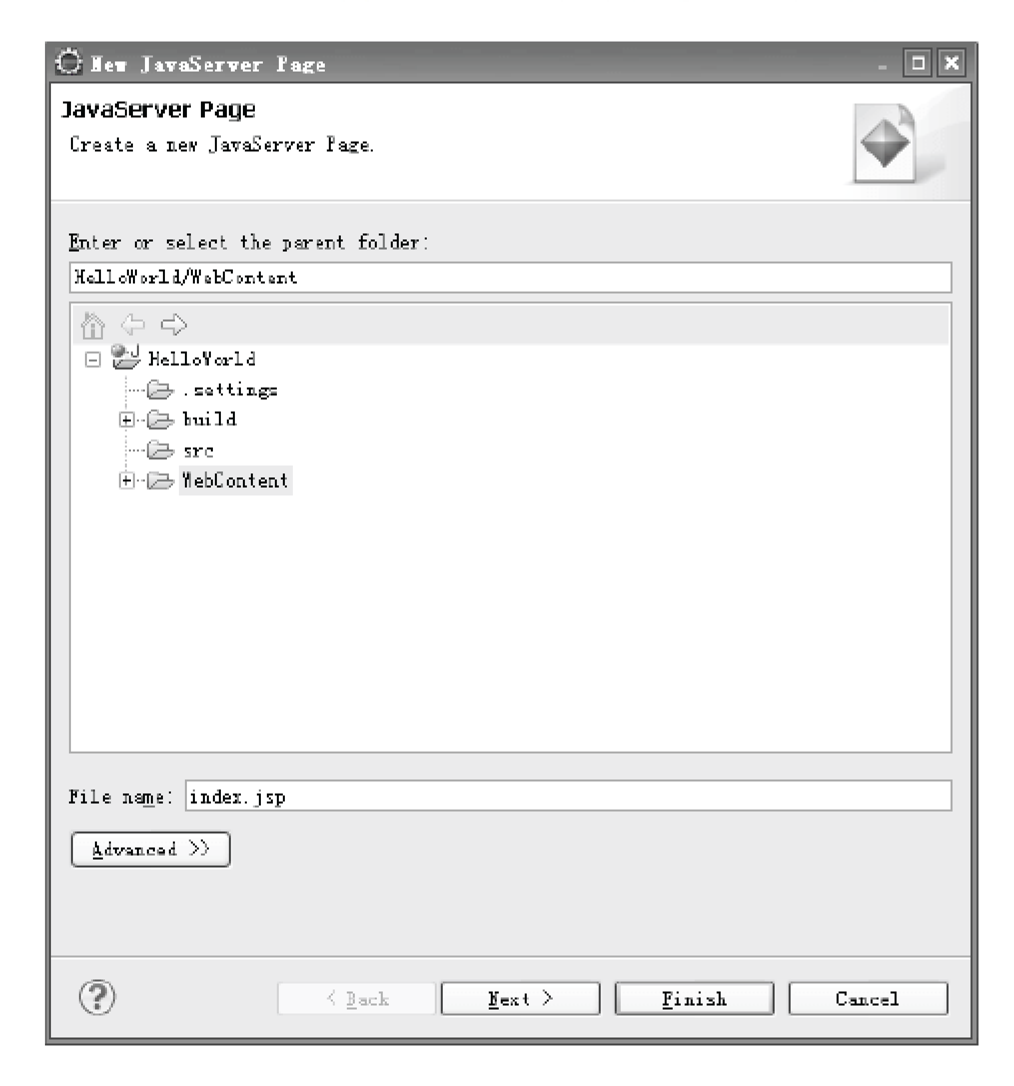


图 7-33　设置 File name

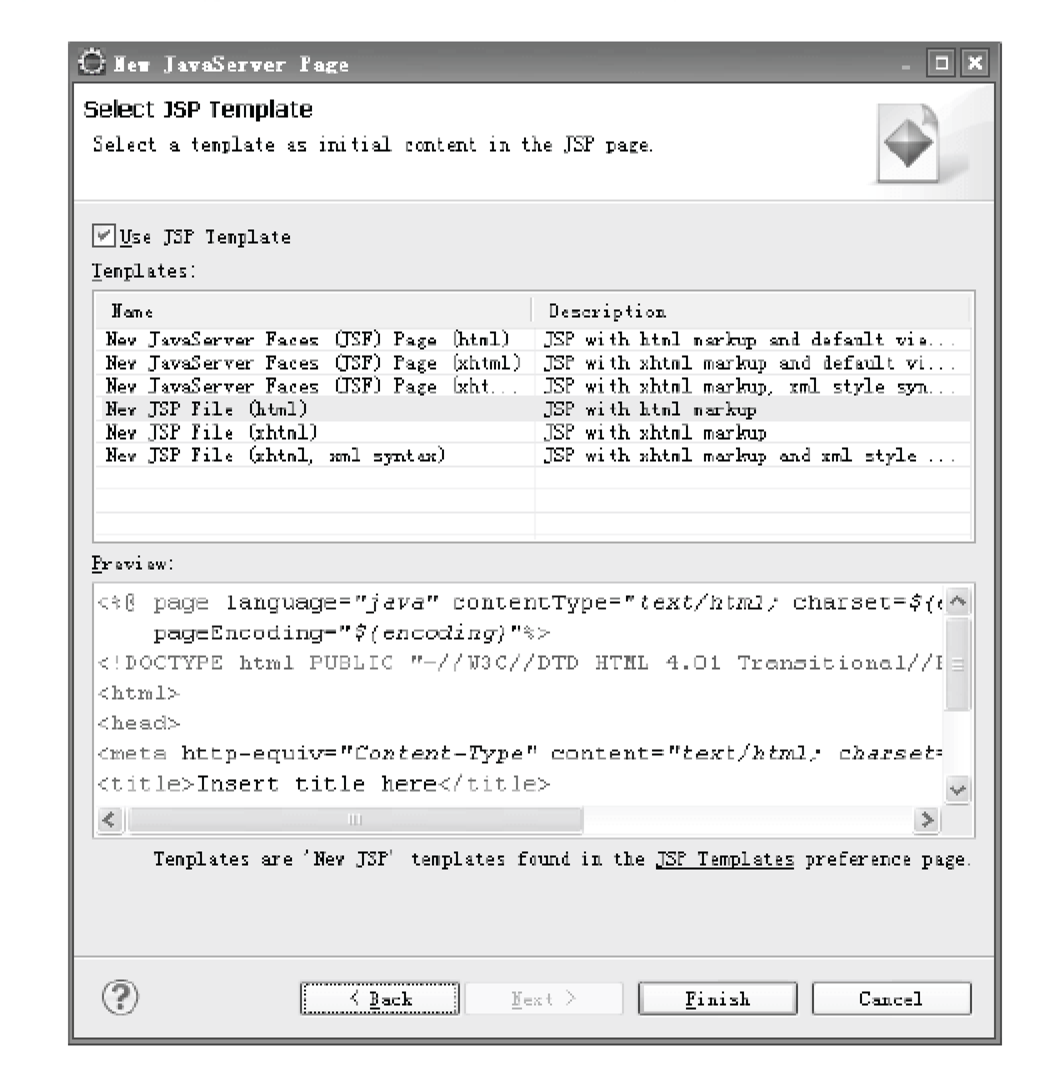


图 7-34　选择 JSP 网页模板

（9）修改 index.jsp，在<body>和</body>标签之间添加下面的代码：

```
Hello,World!<br/>
This is Eclipse and tomcat Server
```

按 Ctrl+S 组合键保存即可，如图 7-35 所示。

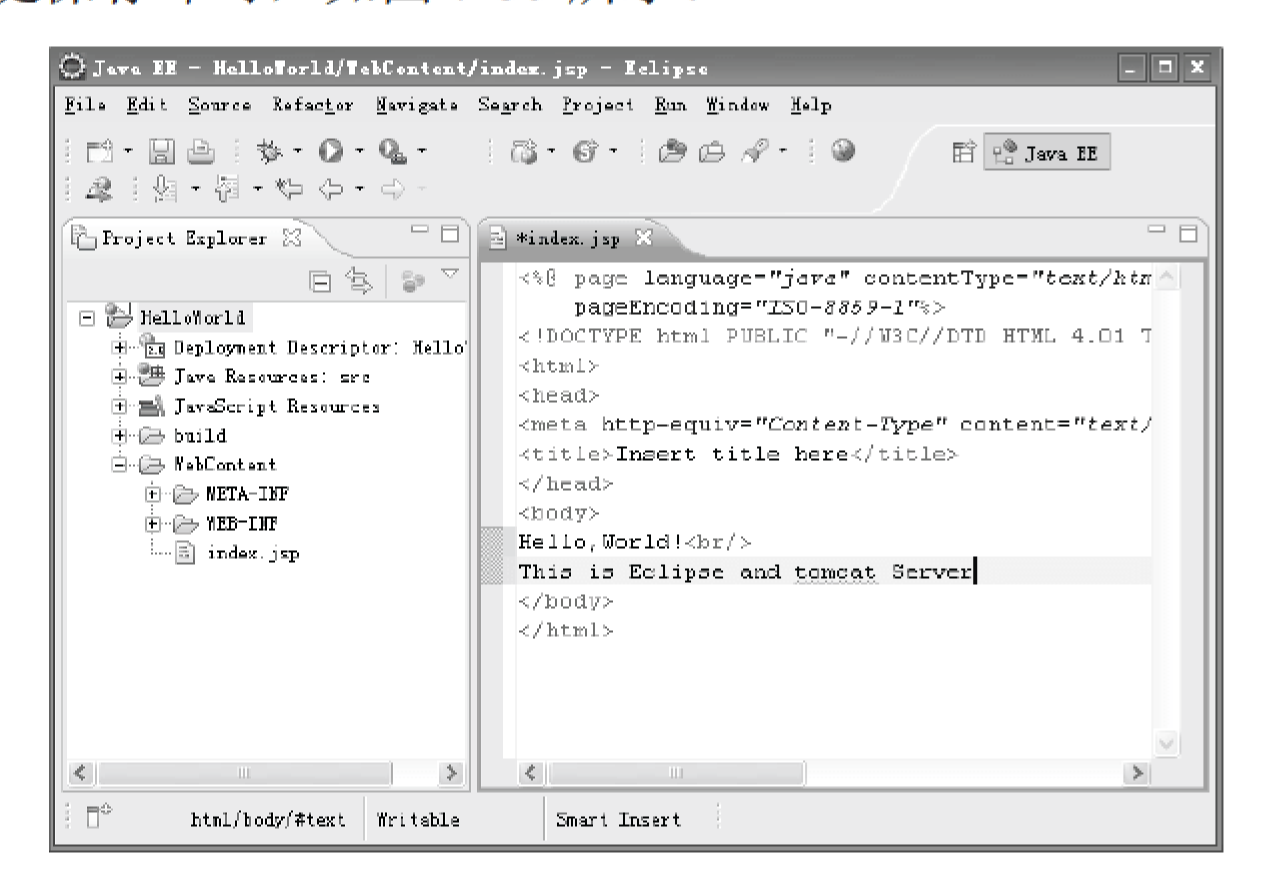


图 7-35　index.jsp 文件

2. 运行工程

（1）在主界面中选择 Run→Run 命令，或者在 HelloWorld 工程上单击鼠标右键，在弹出的快捷菜单中选择 Run As→Run on Server 命令，如图 7-36 所示。

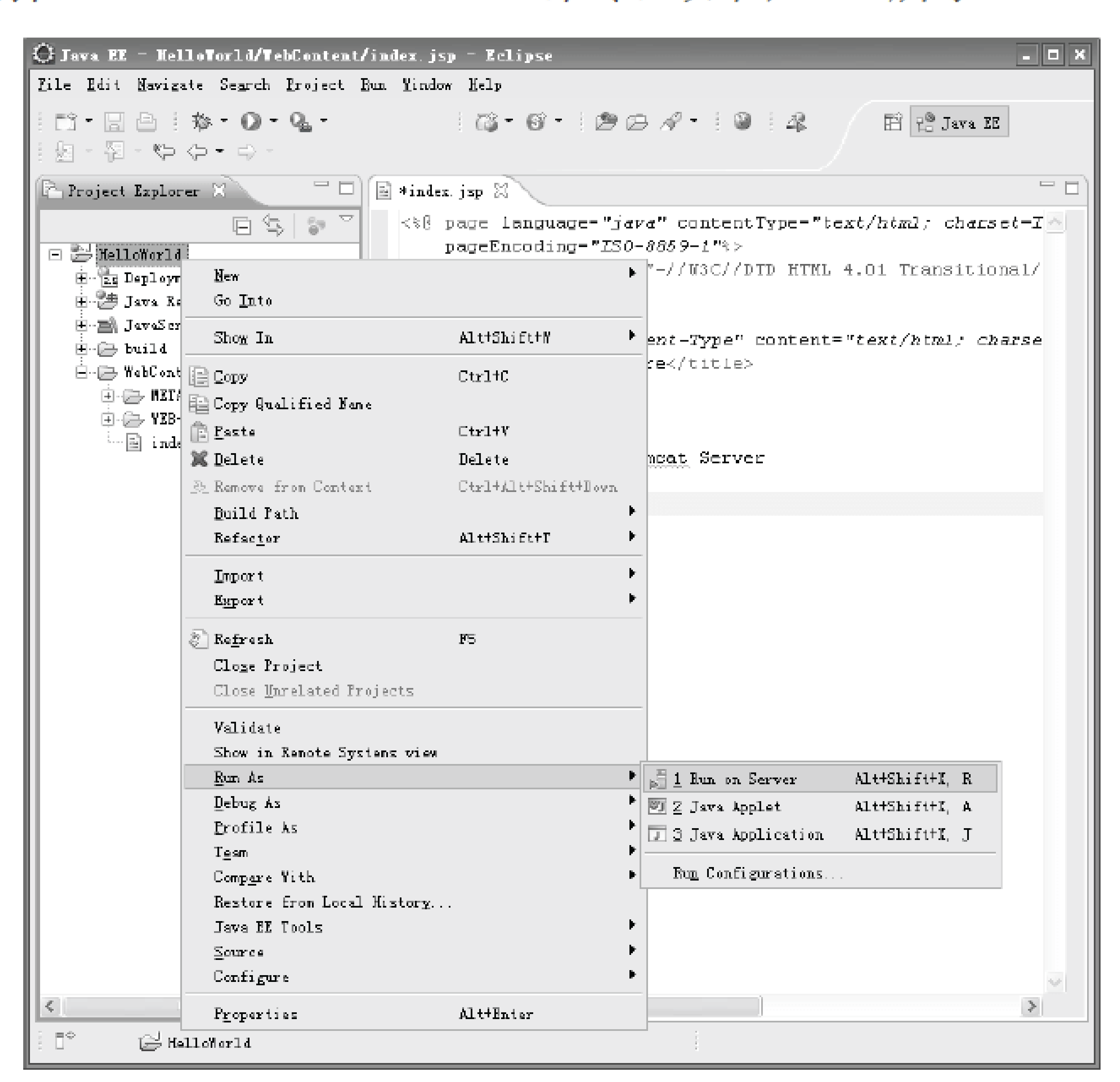


图 7-36　选择命令

（2）弹出 Run On Server 对话框，默认已经选择了 Apache Tomcat v6.0，单击 Next 按钮，如图 7-37 所示。

（3）Run On Server 对话框中已经默认将 HelloWorld 关联到服务器，单击 Finish 按钮结束设置，如图 7-38 所示。



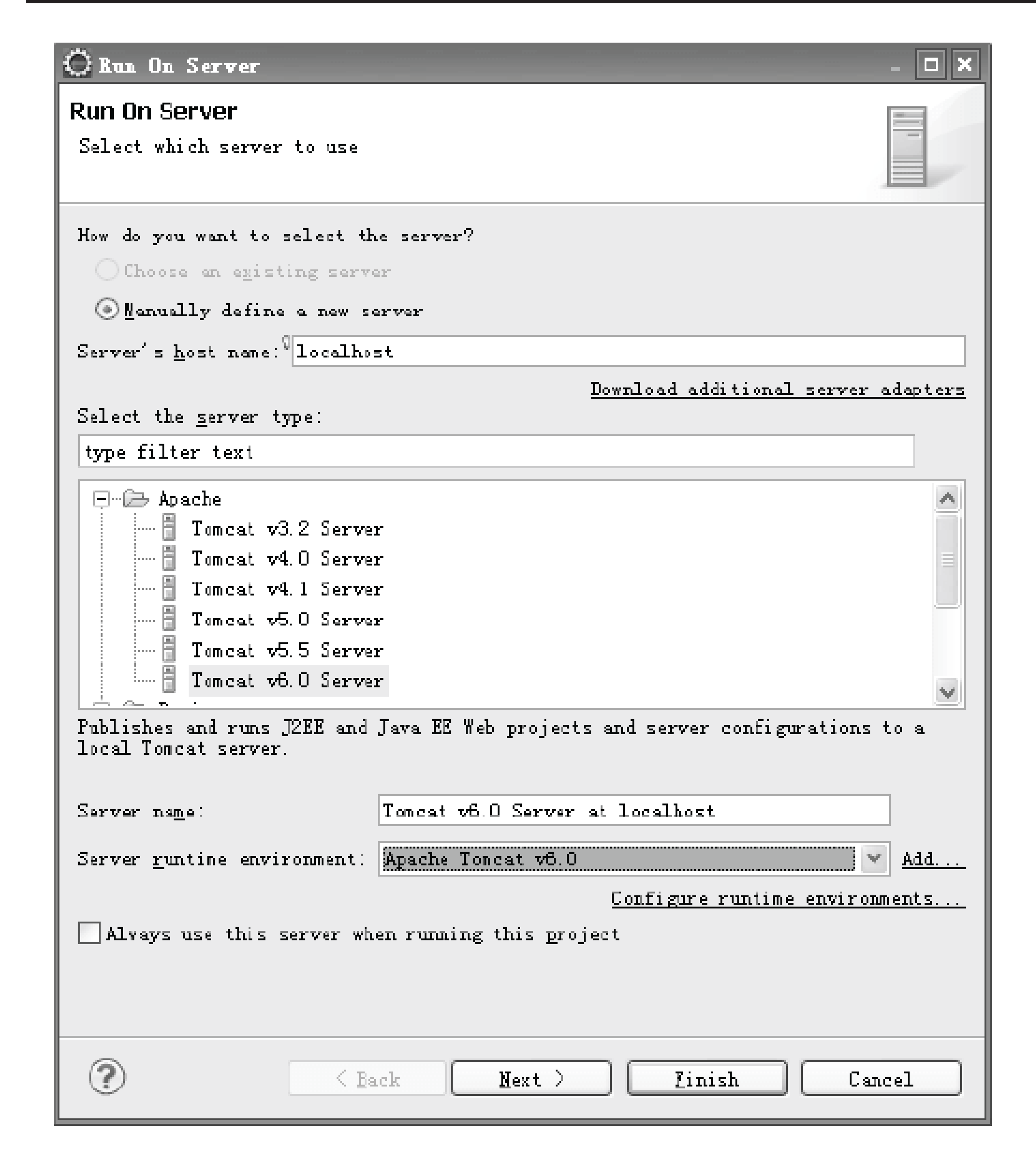


图 7-37　Run On Server 对话框

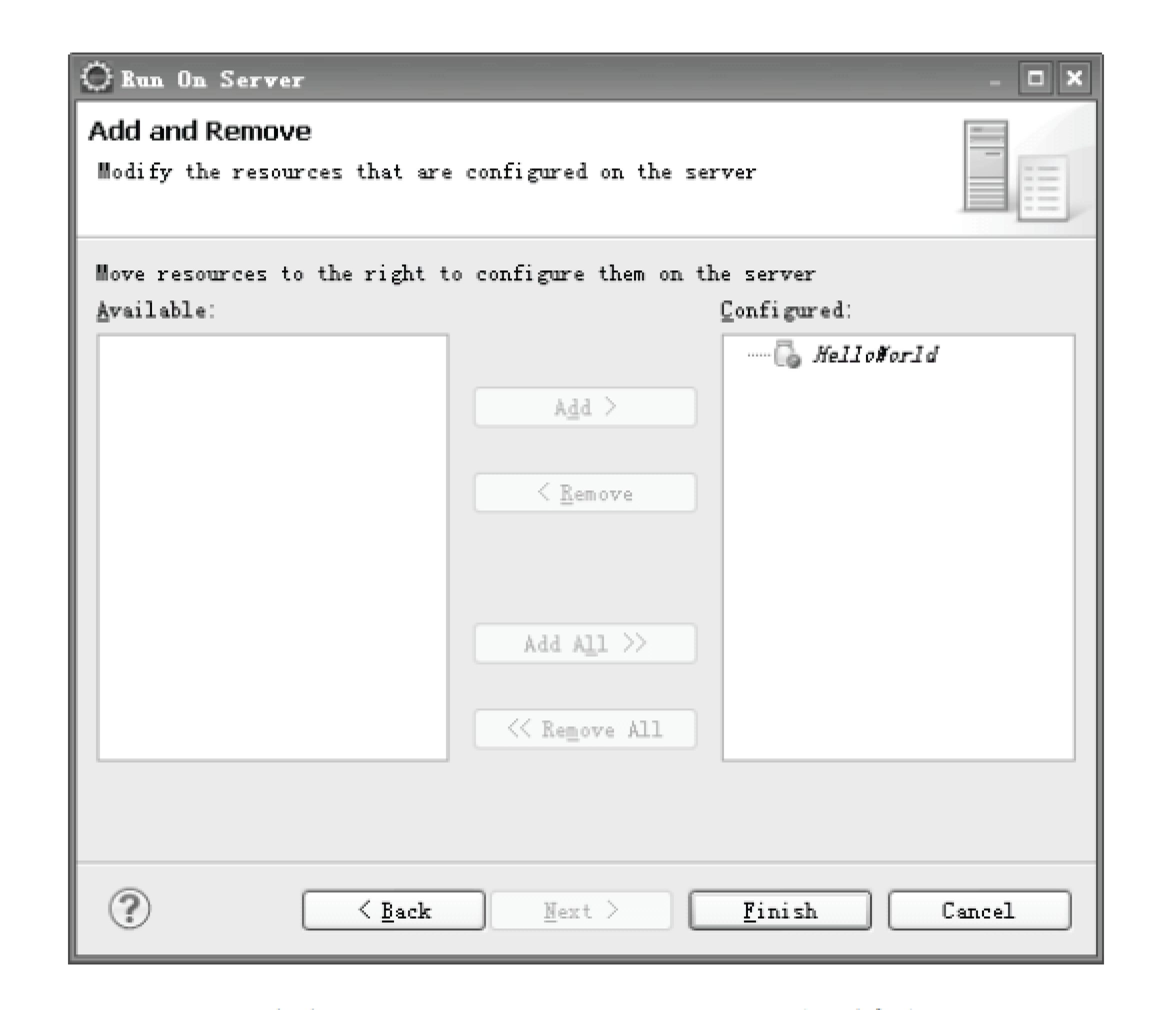


图 7-38　Run On Server 对话框

（4）如果没有什么意外，大约 1 分钟内（Tomcat 启动消耗的时间）就会显示出这个站点，如图 7-39 所示。

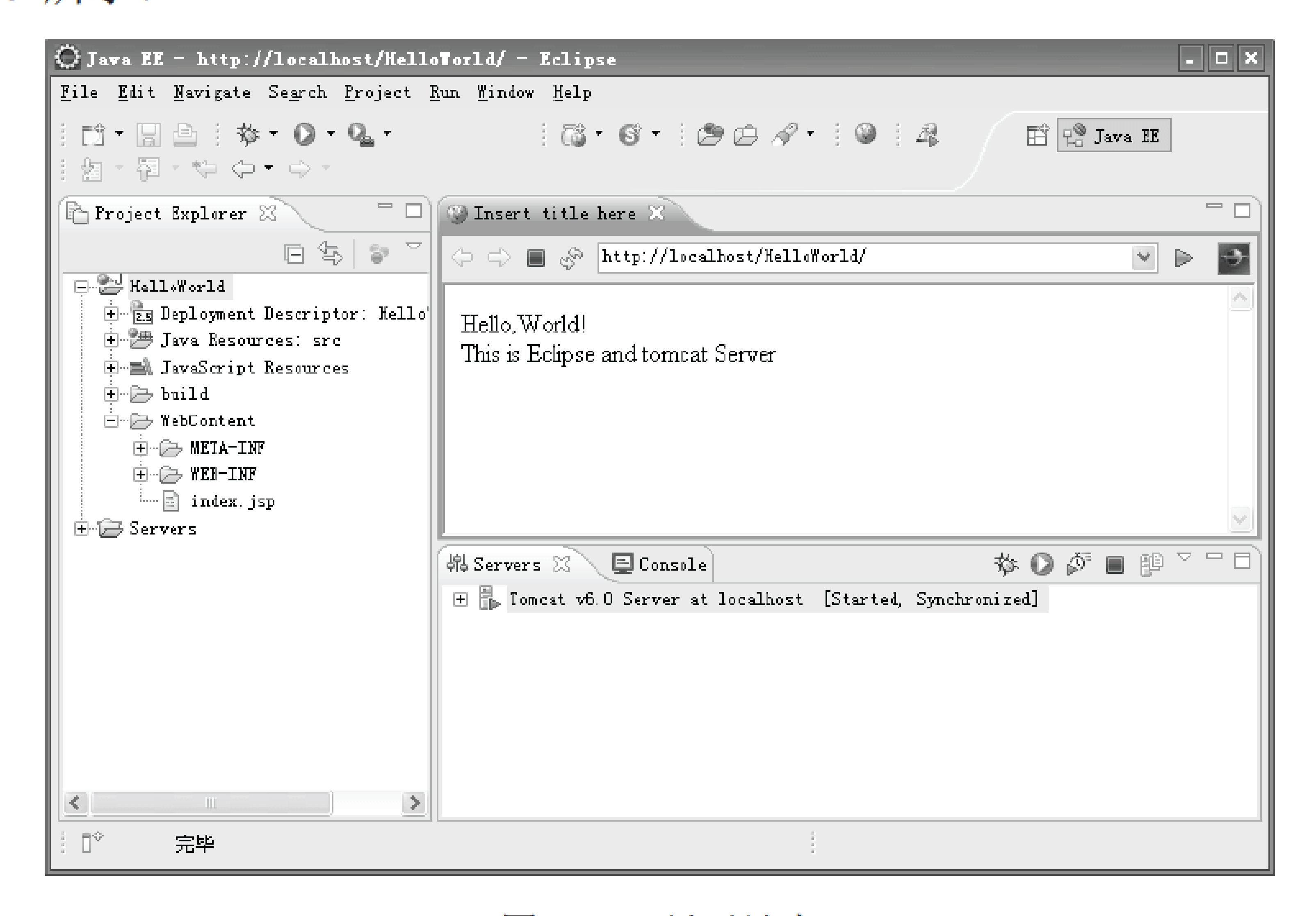


图 7-39　显示站点

3．JSP Web 应用目录结构

在 Tomcat 中部署 Web 应用，需要将开发的 Web 应用包部署到 Tomcat 目录的 webapps 子目录中。应用可以是一个目录，也可以是一个 war 包。当应用为 war 包时，启动 Tomcat 后，Tomcat 会自动将其解压成一个目录。下面以 HelloWorld 为例进行讲解，部署到 Tomcat 的目录为{Tomcat 目录}\webapps\HelloWorld\，则 HelloWorld 拥有以下子目录。

- /WEB-INF：包含应用的配置文件 web.xml 等，其中 web.xml 是必须有的，其他的

根据具体应用安排。

- /WEB-INF/lib：包含 Web 应用所需要的类库文件。
- /WEB-INF/classes：包含 Web 应用所需要的 class 文件。

7.5 JSP 基本知识

要学习 JSP，首先必须了解 Java 的语法。

7.5.1 Java 语法基础

Java 是一种纯面向对象的语言，它有自己的关键词、操作符、表达式和流程控制等，Java 的重点和核心在于类、方法、接口、集成和多态等面向对象的概念，它还有着很好的跨平台性、支持分布式、高度的稳定性和安全性，而且它是编译和解释结合的语言。

Java 从语法上来说是在 C++基础上发展起来的，所以它具有 C 和 C++语言的一些特性，其基本语法与 C 和 C++很相似。

例 7.1 冒泡排序的一个 Java 程序。

```
//Bubble.java
public class Bubble{
    public static void main(String []args){
        int []array={5,8,3,4,9,10,2,1};
        String str="The Result is:";
        int temp;
        for(int j=array.length-1;j>0;j--){
            for(int i=0;i<j;i++){
                if(array[i]<array[i+1]){
                    /*交换两个相邻元素*/
                    temp=array[i];
                    array[i]=array[i+1];
                    array[i+1]=temp;
                }//if
            }//for
        }//for
        System.out.println(str);
        for (int serial=0;serial<array.length;serial++){
            System.out.print(array[serial]);
            System.out.print(",");
        }
 }//main
}//class Bubble
```



在 cmd 窗口中的编译和运行结果如下：

```
C:\Java>javac Bubble.java
C:\Java>java Bubble
The Result is:
10,9,8,5,4,3,2,1,
C:\Java>
```

下面介绍 Java 程序的组成。

1．语法注释

Java 中的注释包括了 C++中的注释，同时还有一种新增加的 Java 注释。

- /* 注释内容 */：在“/*”和“*/”之间的所有内容均被注释，适合于多行的注释。
- // 注释内容：在“//”之后的所有内容均被注释，适合于单行注释。
- /** 注释内容 */：这类注释属于第一种注释，但是在 Java 中，这类注释被特殊处理，出现在任何声明之前时才会被特殊处理。这类注释意味着被括起来的部分应该作为声明项目的描述，可被包含在 Javadoc 程序自动产生的文档中。

2．Java 关键字

关键字是构成编程语言本身的符号，是一种特殊的标识符，又称作保留字，如 class、static、int、for 等都是 Java 的关键字。Java 的关键字共有 40 多个，如表 7-1 所示。

表 7-1　Java 的关键字

abstract	boolean	int	interface
break	byte	long	native
byvalue	case	package	private
cast	class	protected	public
continue	default	return	short
do	double	static	super
false	final	switch	synchronized
finally	float	this	threadsafe
for	goto	throw	transient
if	implements	true	try
import	instanceof	void	while

关键字在 Java 语言中有特殊的含义，所以不能当作一般的标识符使用，即一般的标识符，如变量名、类名、方法名等不能是关键字。true、false 和 null 通常也被看成是关键字。

3．数据类型

Java 数据类型包括两大数据类型，分别为基本数据类型和结构数据类型。

Java 基本数据类型包括整数型、浮点型、字符型、布尔型 4 类，其中整数型和浮点型又可细分为若干种。

- 整数型：Java 定义了 4 种整数类型，如表 7-2 所示。

表 7-2　4 种整数类型

数 据 类 型	比　特　位	字　　节	取 值 范 围
byte	8	1	$-2^{7}\sim2^{7}-1$
short	16	2	$-2^{15}\sim2^{15}-1$
int	32	4	$-2^{31}\sim2^{31}-1$
long	64	8	$-2^{63}\sim2^{63}-1$

- 浮点型：浮点数即实数，包括整数和小数部分，Java 包括两种浮点型，分别为 float 和 double，分别代表单精度和双精度，如表 7-3 所示。

表 7-3　浮点型

数 据 类 型	比　特　位	字　　节	取 值 范 围
float	32	4	$10^{-37}\sim10^{38}$
double	64	8	$10^{-307}\sim10^{308}$

- 字符型：在 Java 中用于存储字符的数据。在 Java 中采用 Unicode 字符集，所以 char 占用 16 位，而不是 C 语言中的 8 位，取值范围为 0～65535。
- 布尔型：布尔型数据类型只有两个，即 true 和 false。布尔型和其他数据类型不能相互转换。所有的关系运算符返回值都是布尔值。
- 在 Java 中，除了布尔型外，都可以混合运算，在运算过程中，不同的数据类型先转换为统一数据类型的数据，然后再运算。

（1）自动类型转换

自动类型转换是从低级到高级发生自动类型转换，从表达范围小的类型向表达范围大的类型发生自动类型转换。优先级如下：

低---→高

byte、short、char　→　int　→　long　→　float　→　double

（2）强制转换

当数据类型由高级向低级转换时，需要强制类型转换。如将一个 double 类型数据转换为一个 int 类型的数据时，就必须使用强制类型转换。如“double d=9.9;int i=(int)d;”，那么 i 的值就是 9。

4．常量

常量是指 Java 程序中其值在程序执行期间不能改变的量。按照类型常量可分为整数类型常量、实数类型常量、字符常量、布尔常量和字符串常量。

（1）整数类型常量

Java 中的整数类型常量又分为 3 种：十进制、八进制和十六进制。例如，12 是十进制常量、0766 是八进制常量（以 0 开头就是八进制）、0xfa1 是十六进制常量（大小写均可）。



（2）实数类型常量

在 Java 中实数类型常量分为单精度和双精度两种类型，如 241.3f 为单精度（在其后加上大小写 F 均可），241.3d 为双精度（在其后加上大小写 D 均可），其区别在于所占空间不同，表示的范围也不同。若没有后缀字母，则系统默认为双精度类型。

实数类型常量也可以使用科学表达式，如 0.2E10 的值和 2.0E9 的值相同，都为 2000000000，其中大小写 E 均可。

（3）字符常量

字符常量就是用单引号括起来的一个字符，如 'b'、'B'。Java 中的字符数据是 16 位无符号型数据，可以使用 Unicode 字符集，格式为 '\uxxxx'，其中 u 必须为小写，xxxx 表示该字符的十六进制的 Unicode 编码值。

（4）布尔常量

布尔常量只有两个：true 和 false，分别表示逻辑运算的真和假。

（5）字符串常量

字符串常量使用双引号括起来的字符串，如"Hello Java"。在 Java 中，字符串常量是作为 String 类的一个对象来处理的，可以使用连接操作符"+"来连接两个字符串组成更长的字符串。

5．变量

变量的命名必须符合一定的规则：

- 由字母、数字、下划线（_）和美元符号（$）构成。
- 由字母、下划线（_）和美元符号（$）开头，不能以数字开头。
- 不能与关键字同名。
- 符合一定的书写规则，如类名首字母大写、变量一律小写字母开头、宏定义全部用大写字母、尽量不要缩写、要用英文不要用拼音等。养成好的书写习惯可以增加程序的可读性，具体的规则因人而异。

变量的定义格式如下：

[修饰符] <类型名> <变量名> [=<初值>] [, <变量名> [=<初值>]…]

其中，括号"[]"代表可有可无，尖括号"<>"代表必须有，嵌套的必须从外向里依次满足条件。Java 中的变量分为整型变量、实数型变量、字符型变量和布尔型变量 4 种，其定义格式分别如下：

```
//整型变量
int i; byte b; short s; long l;
//实数型变量
float f; double d;
//字符型变量
char c='C';
//布尔型变量
Boolean b2=false;
```

6. 转义字符

Java 提供了转义字符，以反斜杠“\”开头，如表 7-4 所示。

表 7-4　Java 中的转义字符

转 义 字 符	描　　述
\ddd	1～3 位八进制
\uxxx	1～4 位十六进制
\'	单引号字符
\"	双引号字符
\\	反斜杠
\r	回车（回到本行开始）
\n	换行（到光标下一行的正下方）
\f	走纸换页
\t	制表符（一般长度为 4 个空格）
\b	退格

7. 分隔符

分隔符是指在语句中用来分隔变量、常量或语句的符号。Java 中的分隔符如表 7-5 所示。

表 7-5　Java 中的分隔符

分隔符符号	含　　义
,	分隔变量
.	用来分开包和子包
;	表示一条语句的结束
()	在方法中调用参数列表；定义表达式优先级； 在控制语句中包含表达式
{}	包含自动初始化数组的值
[]	声明数组

8. 运算符与表达式

运算符是对变量或其他对象进行运算操作的特定符号。Java 语言内部有 44 个运算符，分为 6 类，分别为算术运算符、赋值运算符、关系运算符、逻辑运算符、位运算符和条件运算符。

表达式是由常量、变量、方法调用以及运算符按照一定规则组合而成的，用于计算或者对变量进行赋值。

（1）算术运算符

算术运算符包括“+”、“-”、“*”、“/”、“%”、“++”、“--”，其中，“+”和“-”还有表达正数、负数的意思。“++”和“--”用法较为特殊，分为前缀和后缀两种，如“i++”是后缀的，“--i”是前缀的，其作用是对变量进行“加 1”或者“减 1”操作，前缀是在表达



式中首先进行“加 1”或“减 1”操作，然后再进行表达式其他部分；后缀是在表达式中进行表达式整体运算后，再进行“加 1”或者“减 1”操作。

（2）赋值运算符

赋值运算符“=”就是把右边的表达式的值赋给左边的变量（且左边的变量只能是一个符合变量命名规则的变量），右边表达式可以是常量、变量、表达式，如 b=a+5 是将 a+5 的值赋给变量 b。

如果左、右的数据类型不一致，则需要先对右边表达式进行强制转换才可以赋值，否则编译出错。若在赋值操作符“=”前加上一个单运算符，则构成扩展赋值运算符，如表 7-6 所示。

表 7-6 赋值运算符

运 算 符	含 义	例 子	等 价 效 果
+=	自增加 x	a+=3	a=a+3
−=	自减少 x	a−=3	a=a−3
*=	自乘以 x	a*=3	a=a*3
/=	自除以 x	a/=3	a=a/3
%=	对 x 取模	a%=3	a=a%3

（3）关系运算符

关系运算符是用来对两个操作数进行比较的。关系表达式就是用关系运算符将两个表达式连接起来的一种操作，其运算结果为布尔逻辑值。例如，假设 A=4，B=5，则表 7-7 包括了所有的关系运算符。

表 7-7 关系运算符

运 算 符	含 义	例 子	布 尔 值
>	大于	A>B	false
>=	大于等于	A>=B	false
<	小于	A<B	true
<=	小于等于	A<=B	true
!=	不等于	A!=B	true
==	等于	A==B	false

需要注意的是，关系运算符“==”和赋值运算符“=”很容易混淆，编程人员经常犯的错误之一就是在比较是否相同时少写了一个“=”，结果就成了赋值运算符，这种逻辑错误可能会浪费大量的时间。一种有效的解决办法就是：在进行比较时，将常量放在左边，如“2==a”，这样如果少写了一个“=”，就违背了赋值运算符左侧必须是变量的规则，就会导致编译出错，容易排查错误。

（4）逻辑运算符

逻辑运算符只能处理布尔型的数据，其结果也是布尔值。例如，假设 A、B 均为波尔类型值，则表 7-8 列出了所有的逻辑运算符。

表 7-8　逻辑运算符

运　算　符	含　　义	例　　子	例 子 解 释
&	逻辑与	A&B	当 A、B 均为 true 时，为 true，否则为 false
|	逻辑或	A|B	当 A、B 均为 false 时，为 false，否则为 true
&&	快速逻辑与	A&&B	当 A 为 false 时，为 false，否则为 B
||	快速逻辑或	A||B	当 A 为 true 时，为 true，否则为 B
!	逻辑非	!B	当 B 为 false 时，为 true，否则为 false
^	逻辑异或	A^B	当 A、B 不同时，为 true，否则为 false

注意：快速逻辑与、或在一些情况下不需要对 B 进行运算，而逻辑与、或则必须将 A、B 全部运算才能得出结果。

（5）位运算符

位运算符只对二进制数据进行操作，分为逻辑位运算和位移运算符两类。例如，假设 Op、Op1、Op2 为一些数据（可以是 int、char、long 等），则表 7-9 列出了所有的位操作符。

表 7-9　位操作符

运　算　符	含　　义	例　　子	例 子 解 释
~	按位取反	~Op	对 Op 按位取反
&	按位与	Op1&Op2	使 Op1 和 Op2 按位与
|	按位或	Op1|Op2	使 Op1 和 Op2 按位或
^	按位异或	Op1^Op2	使 Op1 和 Op2 按位异或
>>	按位右移	Op1>>Op2	使 Op1 右移 Op2
<<	按位左移	Op1<<Op2	使 Op1 左移 Op2
>>>	无符号右移	Op1>>>Op2	使 Op1 无符号右移 Op2

逻辑与“&”和逻辑或“|”与位操作符的按位与“&”和按位或“|”很容易混淆，它们的最大区别在于左右两侧的数据类型不相同，逻辑与、或的两侧只能是布尔逻辑值。

（6）条件运算符

条件运算符是唯一的三元运算符，格式如下：

Expression?statement1: statement 2

表达式 Expression 的结果应为布尔型。以上语句的解释为：如果 Expression 的结果是 true，则执行语句 statement1，否则执行语句 statement2。其中 statement1 和 statement2 必须有相同的返回值数据类型，并且该数据类型必须有返回值。下面是一个条件运算符的例子：

```
int A=4,B=3;
String C;
C=(A==B)?"A and B are same":"A and B are different";
System.out.println(C);
```

其结果为 A and B are different。



（7）运算符的优先级

任意一个表达式可能存在多个运算符，优先级高的先运算因此运算符的优先级就显得十分重要。Java 语言的运算符优先级如表 7-10 所示。

表 7-10　运算符的优先级

优　先　级	运算符类型	操　作　符
高	一元操作符	+、−、++、−−（+、−表示正、负）
↓	算术操作符	+、−、*、/、%、<<、>>
	关系操作符	>、<、>=、<=、!=、==
	逻辑操作符	&&、‖、&、|、^
	三元操作符	A>B?X:Y
低	赋值操作符	=

9．数组与字符串

数组是可以存储多个数据的数据结构，是数目固定、类型相同的若干个变量的有序集合，其类型可以是整型数组、字符串数组等。数组可以是一维或多维数组。

1）数组

（1）一维数组的定义及使用

格式如下：

类型说明符 数组名[] 或类型说明符 [] 数组名;

例如：

```
int a[];
String []b;
```

数组定义完毕后，还必须使用 new 关键字为数组分配了空间后才能使用，如“:a=new int[12];b=new String[20]”。此外，在定义数组的同时也可以对其进行分配空间，如“String []b=new String[20];”。

（2）二维数组的定义及使用

格式如下：

类型说明符 数组名[][] 或类型说明符 [][] 数组名;

例如：

```
int a[][];
String [][]b;
```

定义二维数组后，还必须使用 new 关键字为数组分配了空间后才能使用，例如：

```
a=new int[12][13];
b=new String[20][5];
```

在定义数组的同时也可以对其进行分配空间，例如：

```
String [][]b=new String[20][5];
```

2）字符串

Java 提供了两个字符串类，即 String 类和 StringBuffer 类。字符串是作为 String 对象来进行处理的，在字符串对象中封装了一些类的方法和属性来进行字符串处理，这极大地方便了程序的编写，提高了工作效率。下面介绍一些 String 类的常用方法。

（1）字符串的声明

字符串是用双引号括起来的一些 Java 合法的字符。声明方法为：

```
String   s; s="Hello,World!";
```

也可以使用 new String()方法给字符串变量赋值。例如：

```
String s=new String("Hello,World!");
```

（2）字符串的比较

常用的字符串比较的方法有 equals()、equalsIgnoreCase()、startsWith()、endsWith()、regionMatches()、compareTo()、compareToTgnoreCase()等。

① equals 方法

equals 方法的定义如下：

Public Boolean equals(String s)

此方法用来比较当前字符串对象的实体是否与指定的字符串 s 实体相同。例如：

```
String a=new String("I like blue");
String b=new String("I like blue");
System.out.println("a equals b="+a.equals(b));//其结果是“a equals b=true”。
```

② equalsIgnoreCase 方法

equalsIgnoreCase 方法的定义如下：

Public Boolean equalsIgnoreCase (String s)

此方法用来比较当前字符串对象的实体是否与指定的字符串 s 实体相同，比较时忽略大小写。例如：

```
String a=new String("I like blue");
String b=new String("I Like BLUE");
System.out.println("a equals b="+a. equalsIgnoreCase (b));//其结果是“a equals b=true”。
```

③ startsWith 方法

startsWith 方法的定义如下：

Public Boolean startsWith (String s)

此方法用来判断当前字符串对象的前缀是否是参数指定的字符串 s。例如：

```
String a=new String("WelcomeToChina!");
System.out.println("a 的前缀是 Welcome 是:"+a. startsWith("Welcome"));//其结果是“a 的前缀是
//Welcome 是:true”
```



④ endsWith 方法

endsWith 方法的定义如下：

Public Boolean endsWith (String s)

此方法用来判断当前字符串对象的后缀是否是参数指定的字符串 s。例如：

```
String a=new String("WelcomeToChina!");
System.out.println("a 的后缀是 China 是:"+a. endsWith ("China"));//其结果是“a 的后缀是 China
//是:false”，因为还有个感叹号
```

⑤ regionMatches 方法

regionMatches 方法的定义如下：

Public Boolean regionMatches (int firstStart,String other,int otherStart,int length)

或

Public Boolean regionMatches (Bolean ignoreCase,int firstStart,String other,int otherStart,int length)

此方法用来判断从当前字符串的 firstStart 指定的位置开始，取长度为 length 的一个子串，并将这个子串和参数字符串 other 指定的一个子串进行比较。其中 other 指定的子串是从参数 otherStart 指定的位置开始，从 other 中取长度为 length 的一个子串。如果两个子串相同就返回 true，否则返回 false。字符串的位置从 0 开始。例如：

```
String other=new String("WelcomeToChina!");
String me=new String("Welcom");
System.out.println("WelcomeTochina!的第 2+1-----2+1+4 个字符是 lcom 是:"+me.regionMatches
(2,other,2,4));
//结果是：“WelcomeTochina!的第 2+1-----2+1+4 个字符是 lcom 是:true”
//由于参数过多，请仔细体会
//对于第二种定义，又增加了一个是否忽略大小写的选项，默认 false。
```

⑥ compareTo 和 compareToIgnoreCase 方法

compareTo 和 compareToIgnoreCase 方法的定义如下：

Public int compareTo (String s)

Public int compareToIgnoreCase (String s)

compareTo 方法按照字典顺序与参数 s 指定的字符串进行比较。如果当前字符串与 s 相同，则返回 0；如果当前字符串对象大于 s，则返回正值；如果当前字符串对象小于 s，则返回负值。compareToIgnoreCase 方法忽略大小写，其他和 compareTo 方法一致。例如：

```
String aa=new String("abcd");
String bb=new String("Abc");
System.out.println(aa.compareToIgnoreCase(bb));//整数代表大于，负数代表小于，0 代表等于。
//结果为 1
```

（3）字符串的检索

检索指定字符或字符串在字符串中出现的位置。

声明格式如下：

Public int indexOf(int n);

Public int indexOf(int n,int formindex);

Public int indexOf(String s);

Public int indexOf(String s,int formindex);

以上 4 个方法用于在字符串中定位指定的字符和字符串，并且可指定 formindex 来指定匹配的起始位置。如果没有检测到字符或字符串，返回值为-1。例如：

```
String s1=new String("I have a dream!");
int i;//i 可用 System.out.println 方法打印到屏幕
i=s1.indexOf('e');// i=5
i=s1.indexOf('e',9);// i=11
i=s1.indexOf("have");// i=2
i=s1.indexOf("have",1);// i=1
```

类似的还有 lastIndexOf 方法，其用法和 indexOf 一样，但是返回值是提供最后一个字符或字符串的定位位置。例如：

```
i=s1. lastIndexOf ("e",1);//i=11
```

（4）字符串的截取

在字符串中截取子字符串，声明格式如下：

Public String substring(int startPos)

或

Public String substring(int startPos,endPos)

该方法获得一个当前字符串的子字符串，该子字符串是从当前字符串的 startPos 位置开始截取到最后所得到的子串，第二种定义还可以指定结束位置。例如：

```
String str=new String("If you're tired, take a break!");
System.out.println(str.substring(4,5));//小写字母 o
```

（5）字符串的替换

在字符串中替换指定的字符串，声明格式如下：

public String replace(String oldChar, String newChar);

public String replaceAll(String regex, String replacement);

该方法将获得在字符串中用参数 oldChar 指定的字符替换由参数 newChar 指定的所有字符而得到的字符串。而 replaceAll 方法的作用是：使用给定的 replacement 字符串替换此字符串匹配给定的正则表达式的每个子字符串。例如：

```
String str=new String("If you're tired, take a break!");
System.out.println(str.replace("you're","I'm"));//结果为 If I'm tired, take a break!
```

replaceAll 方法支持正则表达式，使其很强大，但是正则表达式需要专门进行学习。其



用法和 replace 方法相同，replaceAll 可完全替代 replace 方法。

（6）字符串的去除头尾部空格

在字符串中截取空格，声明格式如下：

Public String trim();

该方法得到一个字符串对象，该对象是指定去掉前后空格的字符串，例如：

```
String strx=new String("        I am alone   !              ");
System.out.println(strx.trim());//其结果为“I am alone   !”
```

（7）字符串大小写相互转换

字符串大小写转换方法的声明格式如下：

Public String toUpperCase();

Public String toLowerCase();

该方法并不会更改原字符串对象，例如：

```
String str=new String("I Love Swimming!");
System.out.println(str.toLowerCase());//全部小写字母
System.out.println(str.toUpperCase());//全部大写字母
//结果为：
//i love swimming!
//I LOVE SWIMMING!
```

（8）字符串和字符串数组的相互转换

字符串对象转换为字符串数组的声明格式如下：

Public char[] toCharArray();

该方法可将一个字符串转换为字符串数组。例如：

```
String h=new String("hello");
 char[]charArray=h.toCharArray();
 for(int i=0;i<=h.length();i++)
 System.out.printf("%c\n",charArray[i]);//i=0～4
```

10．语句

每种程序语言都是由语句构成的，即语句是构成程序的基本结构单位。语句提供对程序流程的控制。在 Java 语言中，语句以分号结尾，但语句块或语句块结尾的语句则不需要用分号结尾。Java 是通过控制语句来控制程序的执行的。Java 控制语句分为条件语句、循环语句和跳转语句等。

1）条件语句

条件语句允许程序在运行过程中根据其状态控制语句的执行过程，即控制执行满足某些条件的语句。条件语句包括以下几种。

（1）if 语句

if 语句格式如下：

```
if(条件表达式){
        语句 1;
        …;
    }
else{
        语句 2;
        …;
    }…
```

程序先计算条件表达式的值，如果条件值为 true，就执行语句 1；否则执行语句 2。

（2）switch 语句

switch 语句的格式如下：

```
switch <表达式>{
        case <值 1>:<语句 1>;
                break;
        case <值 2>:<语句 2>;
                break;
        …
        case <值 n>:<语句 n>;
                break;
        default:<语句 n+1;
            }
```

switch 语句是多路分支语句，它提供了一种基于表达式的值来使程序执行不同部分的简单方法。在实现具有多重分支的选择时，即可以使用 if 语句，也可以使用 switch 语句，如果使用 if 语句则会出现较多的嵌套层次，程序结构不清晰，可读性差。因此多分支选择一般使用 switch 语句，并使程序可读性好。

2）循环语句

循环语句的作用就是重复执行某一段程序代码，直到满意一定的条件为止。循环语句由循环初始状态、循环体、条件表达式和控制表达式 4 部分组成。在 Java 中有 3 种循环控制语句，分别为 while 语句、do-while 语句和 for 语句。

（1）while 语句的格式如下：

```
while(条件表达式){
        循环体;
}
```

当条件表达式为 true 时执行循环体语句，直到条件表达式为 false 才退出循环，如果条件表达式是 1 或者 true，则执行死循环，直到循环语句中执行了 break 语句才停止循环。

（2）do-while 语句

do-while 语句和 while 语句很相似，其格式如下：



```
do{
        循环体;
} while(条件表达式);
```

do-while 语句和 while 语句的最大区别在于：在条件表达式为 false 的情况下，do-while 最少执行一次循环体，而 while 一次也不执行。

（3）for 语句

for 语句是使用灵活、功能强大的循环语句，它的使用频率很高。for 语句的格式如下：

```
for(初始化部分;条件判断部分;迭代子部分){
        循环体;
}
```

初始化部分用于设置循环变量的初始值；条件判断部分可以是任意形式的布尔表达式；迭代子部分用于更新循环条件。for 语句的执行过程是：当循环启动，先执行一次（仅这一次）初始化部分，再判断条件判断部分，如果是 false，则跳出 for 循环，for 循环完毕；如果判断部分是 true，则执行循环体，然后执行迭代子部分，再判断条件判断部分，依次循环。

注意： for 关键词后的括号“()”中只能有两个分号“;”。如果和初始化部分等价的语句，在 for 语句前就执行了，则初始化部分可省略，但是分号必须保留；如果迭代子部分等价语句在循环体内部，则迭代子部分可省略；如果条件判断部分为空语句，即只有一个分号“;”，则 for 语句进行死循环，直到循环体有 break 或者其他跳出循环的语句为止。

3）跳转语句

跳转语句是将程序的执行跳转到程序的其他部分。Java 提供了 3 种形式的跳转语句：return 语句、break 语句和 continue 语句。这些语句在使用上都有一定的限制，而不能跳转到程序的任何位置。

（1）return 语句

return 语句用来使程序从当前的方法中退出，返回到调用该方法的语句继续执行。return 语句的格式如下：

```
return <表达式>;
```

其中 return 语句可以带返回值，也可以不带。如果使用 return 语句返回一个值，其必须和该方法声明的类型一致。

（2）break 语句

break 语句用于终止当前执行的循环。break 语句的主要用途有 3 个方面：用于 switch 语句中终止语句的执行；用于循环语句中，退出循环；用于标签中断，程序跳转到指定标签声明中的语句块。break 语句的格式如下：

```
break <标号>;
```

一般用法是直接使用 break，而不加标号。可以使用 break 语句强制退出循环，忽略循环体中的任何其他语句和循环条件。在使用时应当注意的是，在一系列嵌套循环中使用 break

语句时，它将仅影响最里面的循环。

（3）continue 语句

continue 语句只能用在循环语句中，其作用是结束本次循环，继续执行下一次循环。continue 语句的格式如下：

continue<标号>;

通常使用 continue 语句不带标号，用于结束本次循环。注意 continue 语句只能用在循环语句中，其作用是结束本次循环，继续执行下一次循环。

11．类和对象

在面向对象编程技术出现之前，计算机编程是面向过程的编程。随着计算机技术的发展，出现了面对对象的编程语言。Java 就是一种完全面向对象的语言，它支持类和对象的概念，自然就支持面对对象的一些重要技术，如抽象、封装、继承和多态等。

（1）面向对象的概念

所谓对象，就是独立存在于世界中的每个实体，即现实世界的每一个实体都是一个具体的对象，如某一个篮球、某一个 DVD 播放机、某一幢房子、某一个苹果等。

所谓类，就是对一组具有共同特性的对象的抽象，如篮球、DVD 播放机、房子、苹果等。

（2）类的定义

类是对象的定义，对象是类的实体。类是 Java 语言的重要成分，学习 Java 语言很重要的内容就是学习和使用 Java 中的类，通过创建类的对象并对其执行相应的操作来实现各种复杂的功能和要求。类定义的主要内容是类的成员变量和成员函数两部分。成员变量定义体现了对这类对象属性的抽象，成员方法定义体现了对这类对象行为的抽象。所有的成员变量和方法定义都包含在花括号中。

类定义格式如下：

```
[修饰符] class 类名 [extends 超类名] [implements 接口名列表]
{
[<成员变量>]…
[<静态初始化>]…
[<构造方法>]…
[<自定义方法>]…
}
```

其中，类修饰符用于说明类的属性，包括 public、abstract 和 final 3 种。一个类可以被多个类修饰符修饰，且修饰符的排列次序无关紧要，但不能有相同的修饰符。3 种修饰符的含义如下。

- public：表示该类为公有类，任何其他包中的类都可以访问它，没有此修饰符的类只能被同一个包内的类访问。
- abstract：表示该类为抽象类，抽象类中包含着一个或者几个没有实现的抽象方法，所以无法创建抽象类的实例。如果其子类不是抽象类，那么这个子类必须实现该抽象类的所有声明的所有的方法。



- final：表示该类为最终类，最终类不能有子类。

类定义中 class 关键字后为类名。为了使类名和一般的成员变量和对象有所区别，通常第一个字母用大写字母表示。

类定义中 extends（继承）关键字用来表明新创建的类继承于某一个类，被继承的类被称为超类，也称作父类。一个类只能继承一个超类。

类定义中 implements（实现）关键字用来表明这个类所实现的接口。

在类定义中用花括号括起来的是类的类体，在类体中定义成员变量、成员方法、构造方法和初始化成员变量等。

例 7.2　类定义示例。

```
/**
*Filename:ClassDemo.java
*@author:***
*/
public class ClassDemo{
        int valuei;
        String values;
        void setValuei(int i){
              valuei=i;
        }
        void setValues(String s){
              values=s;
        }
        void display(){
              System.out.println("valuei="+valuei+",values="+values);
        }
        public static void main(String[]argc){
              ClassDemo iDemo1=new ClassDemo();
              iDemo1.setValuei(20);
              iDemo1.setValues("Hello,World");
              iDemo1.display();
        }
    }
```

（3）类的成员组成

一个类定义中可以有以下 3 种类型的成员。

- 变量：通常类中声明的变量称为域，用来描述类的属性或状态。一个类中的变量既可以是基本数据类型，也可以是其他类的对象。
- 方法：方法是对类中的变量进行操作的可以执行的程序代码序列，定义了该类对象所具有的行为。
- 内部类：定义在一个类中的类。

（4）变量的使用和访问

类中的变量可分为局部变量和成员变量两大类。局部变量是指在方法内部声明的变量，



其作用域是从声明处开始至它所在的语句结束。每当方法被调用时局部变量建立，在方法结束时释放。局部变量在建立时不会赋予初始值，所以在使用之前要对其赋值。成员变量是指类体中但在方法体外声明的变量，其作用域是整个类。相对于局部变量而言，成员变量是类的全部变量。

（5）变量的声明

变量在使用前必须进行声明，声明变量的格式如下：

[<修饰符>] [<类型名>] [<变量名>] [=<初始化表达式>];

修饰符包括 public、private、static 和 final 等。修饰局部变量的修饰符只有 final。由 static 修饰的变量称为静态变量，静态变量也称为类变量，而非静态变量又称为实例变量。

用 final 修饰的变量称为有名常量，与一般常量不同，有名常量必须赋值且只能赋值一次，之后其值就不能再改变。通常可以在定义一个有名常量时包括一个初始化表达式，表达式的值在初始化时被赋给有名常量。由 final 修饰的变量既可以是成员变量也可以是局部变量。成员变量既可以是类变量，也可以是实例变量。

当一个变量用 static 修饰后，它就不属于任何一个类的具体对象，而是被所有对象共享。由于静态变量属于整个类，所以可以通过类名作为前缀访问它。

（6）变量的使用

在一个类中可以访问另一个类的变量或方法。类变量的访问格式如下：

类名.变量（或者方法）

如果创建了某个类的具体对象后，可以通过类名或对象名作为前缀访问静态变量。类对象的访问格式如下：

对象名.变量（或者方法）

（7）变量的初始化

当声明变量时，可以通过常量、表达式或有返回值方法的调用给它们赋值，完成域的初始化，但类型必须正确；也可以在声明时，不给初始化值，而在创建类的对象时通过带有参数的构造方法给它们赋值。

通常，可以在类中定义一个没有参数的构造方法，构造方法中可以没有任何赋值语句，创建对象时调用这个空的构造方法也将把类的变量初始化该类的默认值，例如，整数初始化为 0；实数型初始化为 0.0f、0.0d；逻辑型初始化为 false；字符型初始化为\u0000；类对象初始化为 null，表明引用不指向任何内存地址。

（8）声明其他类对象作为类变量

类的变量可以是基本数据类型，也可以是其他类的对象。如果一个类定义中包含了另一个实例，表示这个类实例具有对另一个实例对象的引用。

12．方法的使用和访问

如果程序中经常用到某些完成相同功能的代码时，就可以把这样的代码段定义为一个函数，即方法，需要实现代码段的功能时就可以调用这个方法。通过定义方法既可以减少程序的代码量，又可以提高程序的模块化程度。



（1）方法的声明

对象具有状态和行为。对象的状态由类定义中的数据成员来表示，对象的行为就由类定义中的方法来描述，通过调用对象方法返回对象的状态。

方法定义的格式如下：

```
[<修饰符>] <返回类型> <方法名> [<参数 1,参数 2,...>]
{
方法体
}
```

修饰符包括 public、private、protected、static、final、abstract、native、synchronized 等。

- 由 static 修饰的方法是静态方法，又称为类方法，而不是实例的方法。类方法可直接通过类名直接调用，而无须创建类实例，正因为如此，在类方法中不能访问类中的实例变量，也不能访问类中的实例方法，只能用类方法调用类变量。
- 用 final 修饰的方法是最终方法，最终方法不能在子类中被覆盖。将一个方法标识为 final 的目的是为了防止子类重新定义继承自父类的方法。
- 用 abstract 修饰的方法是抽象方法，抽象方法只提供一个声明，而不提供方法的实现，即抽象方法只有方法声明，而没有方法体。注意，abstract 和 final 不能同时修饰一个方法。
- 用 native 修饰的方法称为本地方法，本地方法一般由与平台相关的程序设计语言编写。
- 用 synchronized 修饰的方法称为同步方法，用于确保多线程之间的同步。

方法既可以有返回类型，也可以用关键字 void 指明该方法没有返回值。如果返回值类型不是 void，则方法体中应该包含 return 语句，且必须带有表达式；如果返回值类型是 void，则方法体中可以有 return 语句，也可以没有，而且 return 不能有表达式。另外，return 语句必须和返回值类型相同。

（2）方法的调用

调用方法时，在类体中可以直接调用，如果在类体外，可以通过创建类对象引用对象名调用类方法。

方法调用时，实参的数目要与形参的数目相同，实参的数据类型要与形参的数据类型赋值相匹配。

成员变量和方法的作用域是整个类，在类体内成员变量的初始表达式中可以调用方法，在方法的代码体中可以访问变量，而且不受先后次序的限制。

（3）类的构造方法

类的构造方法是类的一种特殊方法，它没有返回值，且名称必须和类名称相同，构造方法的参数用来给类对象的成员变量赋初值。

在有多个构造方法的情况下，一个构造方法可以调用所在类中的另一个构造方法。

可以通过使用类的构造方法来创建对象，如前面的多个实例。构造方法的使用格式如下：

类名 对象名=new 构造方法([参数]);

其中参数可有可无。

13．继承与多态

继承和多态是面向对象语言中非常重要的两个概念。

（1）继承

继承是面向对象技术的一个大特点。子类继承父类之后，就拥有了父类的所有属性和方法；子类可以覆盖父类的方法，也可以对其进行重载（重载是一个类可以有多个相同的方法名，但是参数不同的一种特性）。开发人员可以根据自己的需要在继承类中添加相应的属性和方法，以完成类的进化和升级。

例 7.3　继承定义示例。

```
//父类
//Parent.java
public class Parent {
        private String name=null;

        public String getName(){
                return name;
        }
        public void setName(String s){
                name=s;
        }
    }

//子类
//Children.java
public class Children extends Parent{
    public static void main(String[]argc{
        Children i=new Children();
        Children.setName("LeLe");
        System.out.println(Children.GetName());
    }
}
```

父类提供了一个属性 name，但是无法直接使用方法，父类提供了两个方法可以读取和设置 name。子类又继承了父类，得到了父类的方法，因此可以直接使用父类的方法。

当子类继承父类时，子类的方法对父类方法的处理有覆盖和重载两种。覆盖是指子类的方法与父类的方法名、参数以及返回值完全一致。重载是指子类的方法只和父类的方法名相同，其他参数和返回值不一致。

在 Java 中，子类只能继承一个父类，而不能多继承，但是 Java 的类可以实现多个接口。



（2）多态

多态是面向对象技术的另一个重要特征。多态就是对重载和重写的利用。重写发生在子类，意思就是子类重写父类相同名称的方法。多态是表现多种形态的能力，也可以叫做动态绑定，主要起消除类型间耦合关系的作用。多态性可以是类的构造方法，也可以是成员方法，其主要表现为方法名称及返回值相同，但方法的参数个数及参数类型可以不同，这样就可以针对不同的情况，来解决不同的问题。

例 7.4　多态定义示例。

```
//superA.java
//定义超类 superA
class superA
{
        int i = 100;
        void fun(){
              System.out.println("This is superA");
        }
}
//subB.java
//定义 superA 的子类 subB
class subB extends superA{
      int m = 1;
      void fun(){
      System.out.println("This is subB");
      }
}
//subC.java
//定义 superA 的子类 subC
class subC extends superA{
      int n = 1;
      void fun(){
      System.out.println("This is subC");
      }
}
//Test.java
class Test
{
        public static void main(String[] args){
        superA a;
        subB b = new subB();
        subC c = new subC();
        a=b;
        a.fun();
        a=c;
```

```
            a.fun();
        }
    }
```

运行结果为：

```
This is subB
This is subC
```

上述代码中，subB 和 subC 是超类 superA 的子类，在类 Test 中声明了 3 个引用变量 a、b、c，通过将子类对象引用赋值给超类对象引用变量来实现动态方法调用。为什么结果不是“This is superA”呢？是因为 Java 的这种机制遵循一个原则：当超类对象引用变量引用子类对象时，被引用对象的类型而不是引用变量的类型决定了调用谁的成员方法，但是这个被调用的方法必须是在超类中定义过的，也就是说被子类覆盖的方法。

所以，不要被例 7.4 中 a.fun()所迷惑，虽然写成 a.fun()，但是由于 a 被 b 赋值，指向了子类 subB 的一个实例，因而 a.fun()所调用的 fun()实际上是子类 subB 的成员方法 fun()，它覆盖了超类 superA 的成员方法 fun()；同样，第 2 个 a.fun()调用的是子类 subC 的成员方法 fun()。

另外，如果子类继承的超类是一个抽象类，虽然抽象类不能通过 new 操作符实例化，但是可以创建抽象类的对象引用指向子类对象，以实现运行时多态性，具体的实现方法同例 7.4。

不过，抽象类的子类必须覆盖实现超类中的所有抽象方法，否则子类必须被 abstract 修饰符修饰，当然也就不能被实例化了。

7.5.2 JSP 语法基础

JSP 是一种基于 Java 技术的 Web 开发语言。JSP 页面与传统的 HTML 语言有些类似，也由一些标签组成。所不同的是，它是在服务器端动态生成 Web 页面的技术。JSP 基本语法是基于 Java 语法基础的，但 JSP 作为一个动态网页脚本语言还具有自己的语法规则，如 JSP 标记、JSP 注释的运用、JSP 变量的声明和方法的声明等。可以将 JSP 看作是 Servlet 的延续技术，或者这样描述 JSP 和 Servlet：Servle 是含有 HTML 代码的 Java 程序，而 JSP 是含有 Java 代码的 HTML 程序。

1. JSP 页面的基本结构

在传统的网页中加入 Java 程序片段和 JSP 标记就构成了 JSP 网页。一个 JSP 页面主要由标记、注释、指令、脚本元素、动作元素等内容构成。JSP 页面的基本结构如下。

- 标记：HTML 标记、JSP 标记。
- 注释：HTML 注释、JSP 注释、JSP 隐藏注释。
- 指令：page、include、taglib。
- 脚本元素：变量声明、方法声明、表达式。



- 动作元素：include、usebean、forward、plugin。

例 7.5　一个简单的 JSP 页面示例。

```
<%@page contentType="text/html;charset=gb2312" language="java" %>
<%@page import="java.util.*"%>
<html>
        <head>
              <title>JSP 简单页面结构</title>
        </head>
        <body bgcolor="#FFFFFF">
        这是 HTML 内容
        <p/>
        <%
              /*这是 JSP 注释*/
        %>
        <%!String showDate()//声明方法
        {
              String currentDate;//声明变量
              currentDate=new Date().toString();
              return currentDate;
        }
        %>
        <%!int sum;%>
        <%
              sum=0;
              for(int i=1;i<100;i++)
                    sum+=i;
        %>
        <!--这是 html 注释-->
        <p/>
        <font size="3" color="blue">自然数 1 到 100 的和为：<%=sum%></font>
        <p/>
        <font size="3" color="green">现在的日期时间为：<%=showDate()%></font>
 </body>
</html>
```

第 1 行“<%@page contentType="text/html;charset=gb2312" language="java" %>”表示 JSP 程序代码是以“<%...%>”作为标记的，并且 JSP 以 page 指令开始。page 用来定义 JSP 文件中的全局属性，如设置 JSP 的中文汉字编码字符为“gb2312”、JSP 编程语言为“Java”。第 2 行“<%@page import="java.util.*"%>”表示导入 JSP 页面程序代码所需要的类。在本段 JSP 代码中获取日期时间的方法 Date()属于 java.util 类，所以 JSP 页面开始时要导入 java.util 类，也可以在 JSP 开始不导入 java.util 类，而在程序代码使用的方法名前加上该方法的类名，效果相同。

在这个 JSP 页面还有 JSP 脚本标记符、JSP 注释、Java 注释、html 注释等，其运行结果如图 7-40 所示。



图 7-40　运行结果

2．JSP 注释

在编写应用程序代码时，为了提高程序的可阅读性和易维护性，在程序代码适当位置增加一些注释是非常有必要的。

JSP 页面程序代码中的注释包含 3 种类型：JSP 隐藏注释、HTML 注释和 Java 注释。

- JSP 隐藏注释：用于描述 JSP 程序代码，在 JSP 程序代码中，它不会被 JSP 引擎出力，也不会在客户端的 Web 浏览器中显示。JSP 隐藏注释的格式如下：
  <%--注释文本--%>
- HTML 注释：主要用于在客户端动态显示一个注释信息，即 HTML 注释可以在浏览器源文件窗口中查看到。HTML 注释的格式如下：
  <!--HTML 注释文本-->
- 作为基于 Java 的脚本语言，JSP 可以遵循 Java 语言本身的注释规则对程序代码进行注释，即使用 Java 的 3 种注释——“//注释内容”、“/*注释内容*/”、“/**JavaDoc 注释*/”。JSP 页面的 Java 注释不会在客户端源代码中显示一个注释信息，其格式如下：

```
<%
//注释内容
/*注释内容*/
/**
*JavaDoc 文档注释
*/
%>
```

3．JSP 变量、方法和表达式的使用

JSP 作为基于 Java 的脚本语言，可以遵循 Java 语言本身的语法规则来使用变量、方法和表达式。单 JSP 作为一种动态网页设计语言，同样有自己的语法特点。

在 JSP 页面声明的变量为局部变量，只能用于本页面。JSP 中的声明变量类型可以是基于 Java 的所有变量类型，其语法格式如下：

<%!变量名[=初始值], 变量名[=初始值]…;%>

在 JSP 页面中，方法的声明和变量的声明类似，其语法格式如下：



```
<%!<返回值类型> <方法名称>{
方法体;
}
%>
```

JSP 表达式用于向客户浏览器输出网页信息，它表示 Java 的表达式，不以分号“;”作为结束符。其语法格式如下：

```
<% =表达式; %>
```

4. JSP 指令

JSP 指令（Directive）用于描述 JSP 文件所能执行的 Java 语句的控制信息，对 JSP 页面进行全局或局部的控制。一个 JSP 页面可以声明多个 JSP 指令，每个 JSP 指令可以声明多个属性。JSP 指令的格式如下：

```
<% @DirectiveName attribute="value" %>
```

JSP 定义了以下 3 种不同的指令。

- page 指令：定义与 JSP 页面相关的属性，并和 JSP 引擎进行数据通信。
- include 指令：定义了 JSP 页面需要插入的资源。
- taglib 指令：定义 JSP 页面可以调用的客户标记库。

（1）page 指令

page 指令用来定义 JSP 文件页面的全部信息，及设置 JSP 页面的全部属性，其作用域为 JSP 页面及其包含的文件。page 指令的语法格式如下：

```
<%@page
[language="Java"]
[extends="package.class"]
[import="package.class|package.*"]
[session="true|false"]
[buffer="none|sizekb"]
[autoFlush="true|false"]
[isThreadSafe="true|false"]
[info="text"]
[errorPage="realativeURL"]
[contentType="text/html;charset"]
[isErrorPage="true|false"]
[isELIgnored="true|false"]
%>
```

使用 page 指令应注意以下几个方面：

- page 指令的作用范围为整个 JSP 页面。
- page 指令可以放在 JSP 页面任何位置，无论把它放在 JSP 文件的哪个地方，它的

作用范围都是整个 JSP 页面。为了便于程序的阅读，通常把 page 指令放在 JSP 页面的开始部分。

- 可以在一个 JSP 页面中使用多个 page 指令，但除了 import 属性只能使用一次外，其他属性可以使用多次。
- page 指令的作用域为整个 JSP 页面和它所包含的文件，但该指令不能作用域动态的包含文件，如<jsp:include>。

下面介绍 page 指令中属性的含义和用法。

- language：该属性定义了 JSP 页面中所使用的脚本语言。目前 JSP 必须使用 Java，所以该属性的默认值为 Java。该属性必须设置在任何脚本之前。
- extends：表明 JSP 编译时需要加入的 Java Class 的全名，该属性将限制 JSP 引擎提供的某些超类，这些超类可能会改善所提供服务的品质，因此在使用该属性时必须谨慎。
- import：表明 JSP 页面脚本环境所需要使用的某些类的类型。该属性是 page 页面唯一可以多次设置的属性。
- session：指明 JSP 页面是否插入一个 HTTP 会话，默认值为 true。
- buffer：设置 JSP 网页的缓冲区大小，它的默认值为 8KB，属性值为 none，表示不缓冲。所有的响应都将通过 ServletResponse 的 PrintWriter 输出。
- autoFlush：设置 JSP 页面缓冲区是否自动刷新。它的默认值是 true，表示当前缓冲区满时，到客户端的输出将自动刷新。如果该属性值为 false，当缓冲区满时，将会出现缓冲区溢出异常。如果把 buffer 设置为 none，就不能把 autoFlush 设置为 false。
- isThreadSafe：设置 JSP 页面是否支持线程。默认值为 true，表示在运行 JSP 页面时，会同时接受多个客户的请求。如果该属性值为 false，JSP 引擎将逐个接受客户的请求。
- info：设置页面的文本信息。通过定义一个字符串，用来表示 JSP 页面的说明信息。可以通过 Servlet.getServletInfo()方法获得该属性值。
- errorPage：设置 JSP 页面发生异常时调用的 JSP 页面。当页面出现一个没有被捕捉的异常时，调用错误信息网页，该属性值的默认值为空，即忽略错误信息页面。
- contentType：定义 JSP 页面的字符编码以及响应的 MIME 类型。默认 MIME 类型为 text/html，默认字符集为 ISO-8859-1。
- isELIgored：用来设置 JSP 页面的 EL（表达式语言）是否被忽略。如果取值为 true，则忽略 EL 表达式的计算，反之则不忽略。该属性的默认值将依据 web.xml 描述文件的版本而确定，Servlet 2.3 以前的版本忽略该默认值。

例 7.6　page 指令的应用示例。

```
<%@page contentType="text/html;charset=gb2312" language="java"
import="java.util.*"
session="true"
buffer="16kb"
autoFlush="true"
```



```
isThreadSafe="true"
info="显示现在的时间"
errorPage="error.jsp"
isErrorPage="false"
isELIgnored="false"
%>
<html>
    <head>
        <title>page 指令的应用</title>
    </head>
    <body bgcolor="#FFFFFF">
        page 指令的应用
        <p/>
        <%!String showDate()//声明方法
        {
                String currentDate;//声明变量
                currentDate=new Date().toString();
                return currentDate;
        }
        %>
        <font size="3" color="red">现在的日期时间为：<%=showDate()%></font>
    </body>
</html>
```

结果如图 7-41 所示。

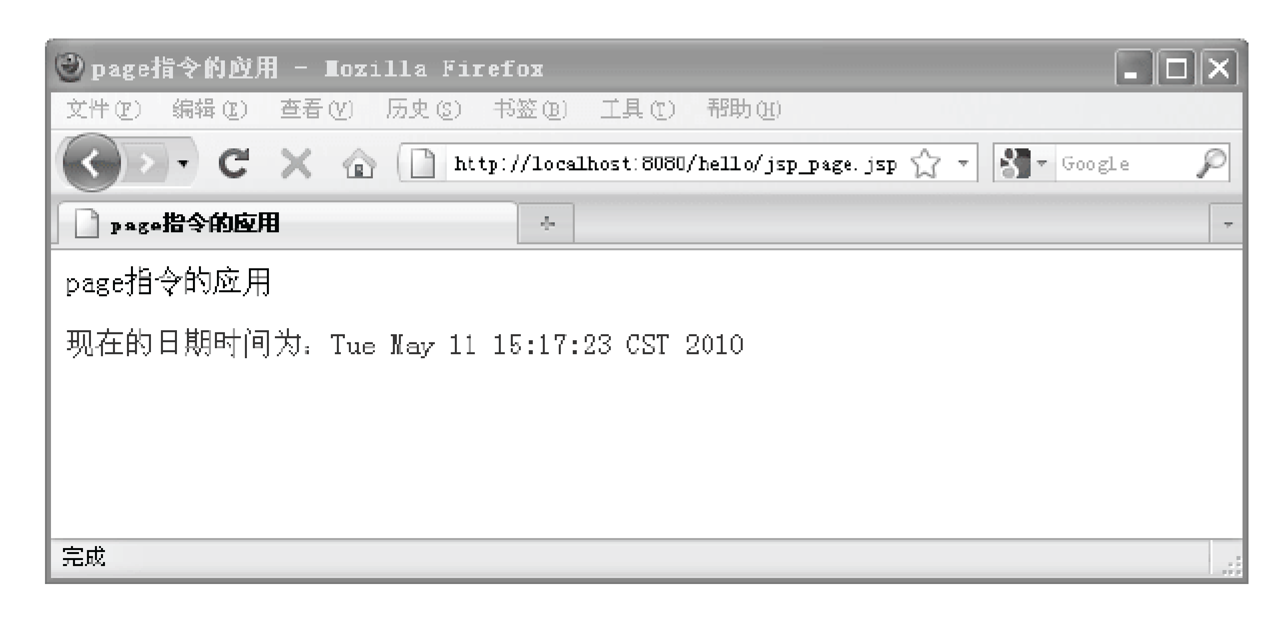


图 7-41　运行结果

（2）include 指令

include 指令用来指定 JSP 页面需要插入的资源，这个资源可以是文本、代码、HTML 文件或 JSP 文件。include 指令所包含的文件必须是静态的文件，而不是包含动态文件。该指令的语法格式如下：

<%@include file =fileName %>

其中，fileName 指被包含的文件的名称，这个文件名称还应包含文件的相对路径。如果路径直接以目录名或文件名开头，则表示该路径是正在使用的 JSP 文件的当前路径；如果路径以“/”开头，表示该路径是参照 JSP 应用的上下文关系路径。

使用 include 指令应该注意的是，include 指令包含的是静态文件，如果在包含的文件中

都设置了 page 指令的 contentType 属性，JSP 引擎将报错。

使用 include 指令有助于实现 JSP 的模块化管理，即可将一个复杂的 JSP 页面划分为若干个部分，以方便 JSP 页面的设计和管理。通常在进行 JSP 页面编写时，将一个 JSP 页面分为 3 个部分：页头（head）、页体（body）、页尾（tail）。响应的可以将一个 JSP 页面分为几个不同的 JSP 页面：head.jsp、body.jsp 和 tail.jsp，当然这只是一个例子。通过 include 指令就可以进行页面合成。

例 7.7　include 指令的应用示例。

```
<%/*head.jsp*/%>
   this is menu bar...

<%/*tail.jsp*/%>
   this is root bar..

<%/*body.jsp*/%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
     pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/ html4/
loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>include test</title>
</head>
<body>
      <table align="center" bgcolor="#ffffff" border="1" bordercolor="red">
      <tr>
      <td>
      <%@include file="head.jsp" %>
      </td>
      </tr>
      <tr>
      <td>
      <br/><br/>
      this body content...
      <br/><br/><br/>
      </td>
      </tr>
      <tr>
      <td>
      <%@include file="tail.jsp" %>
      </td>
      </tr>
      </table>
```



```
</body>
</html>
```

运行结果如图 7-42 所示。

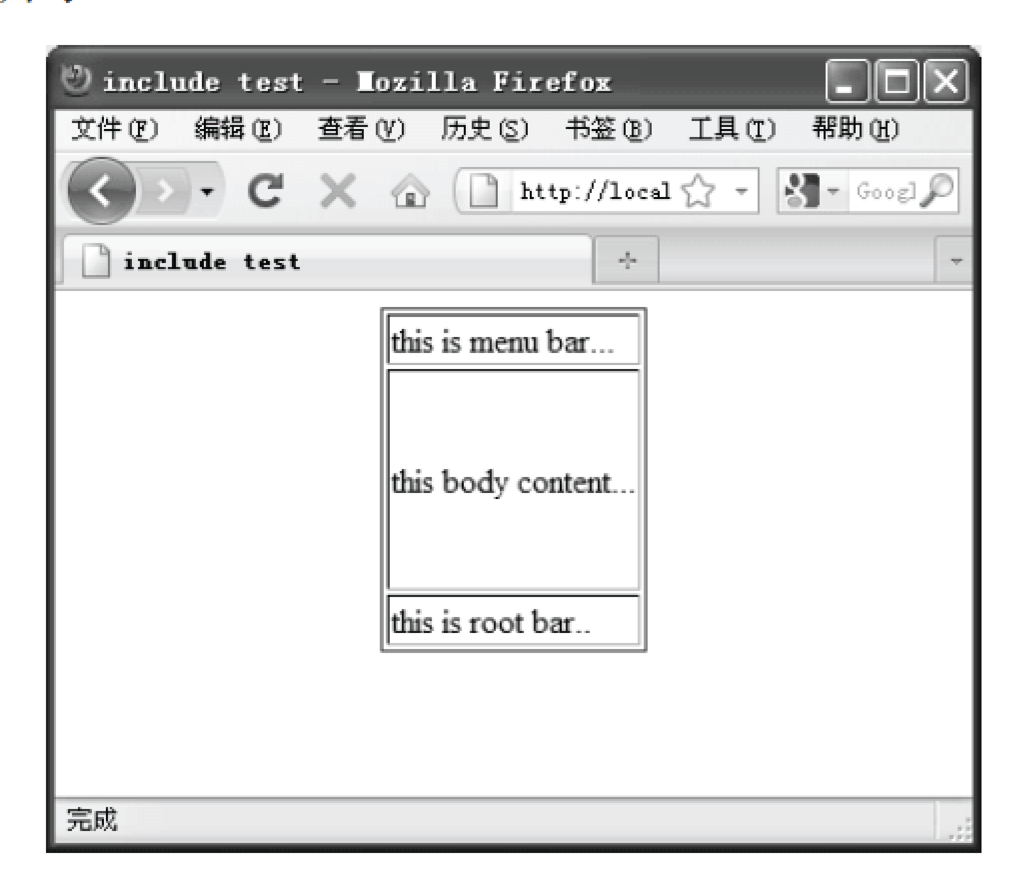


图 7-42　body.jsp 页面的运行结果

（3）taglib 指令

taglib 指令用来定义一个标签库以及自定义标签的前缀，可以把一些需要重复显示的内容自定义为一个标签，以增加程序代码的重用性，方便页面的维护。该指令的语法格式如下：

<%@taglib uri= " tagLibraryURI " prefixx= " tagPrefix " %>

其中，属性 uri（统一资源标识符）用来确定标签库的路径；属性 prefix 定义了 JSP 页面使用此标签库的前缀，标签指令可以在一个页面中多次使用，但其前缀只能使用一次，注意，不能使用 JSP 中的 jsp、jspx、java、javax、servlet 和 sunw 保留字作为前缀。

可以通过前缀来引用标签库中的标签。下面是一个简单的使用 taglib 的例子：

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<c:set var="name" value="hello" />
```

上面代码通过<c:set>标签将 hello 赋值给了变量 name。

5．JSP 动作执行语句

JSP 动作是指 JSP 利用 xml 语法格式的标记来进行 JSP 页面的处理和控制。JSP 很多页面处理都是通过 JSP 中的动作元素来完成的。JSP 动作可以动态地插入文件，重用 JavaBean 组建，重定向页面，从而完成对象的创建和使用。JSP 规范定义了一些标准的动作类型。常用的 JSP 动作如下。

- <jsp:include>：在页面被请求时引入一个文件。
- <jsp:useBean>：引用一个 JavaBean 文件。
- <jsp:setProperty>：设置 JavaBean 属性。
- <jsp:getProperty>：获取 JavaBean 属性。
- <jsp:forward>：把请求转到另外一个文件。

- <jsp:plugin>：根据浏览器类型为 Java 插件生成 OBJECT 或 EMBED 标记。

另外，还有许多动作，如<jsp:param>、<jsp:params>、<jsp:attribute>、<jsp:body>、<jsp:invoke>、<jsp:doBody>、<jsp:element>、<jsp:text>、<jsp:output>、<jsp:declaration>、<jsp:root>、<jsp:scriptlet>和<jsp:expression>等。

下面介绍常用的 JSP 动作元素。

（1）<jsp:include>动作

<jsp:include>动作用来在 JSP 页面插入文件，可以在 JSP 页面中插入静态页面和动态页面。其语法格式为：

<jsp:include page="relativeURL|<%=expression%>" flush="true|false" />

或者向被包含的动态页面中传递参数：

<jsp:include page="relativeURL|<%=expression%>" flush="true|false">
<jsp:param name="参数名称" value="参数值" />
</jsp:include>

其中，“relativeURL|<%=expression%>”为相对路径，可以是字符串，也可以是一个表达式。参数“flush="true"”表示参数名，paramValue 表示参数值。

例 7.8　</jsp:include>动作应用示例。

```
<%/*文件名=jsp_include_Demo.jsp*/%>
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>jsp:inlcude 动作示例</title>
</head>
<body>
<%!String showDate()
{
        String currentDate;
        currentDate=new java.util.Date().toString();
        return currentDate;
}
%>
<center>
        <p>Time Now is :<%=showDate() %></p>
        <jsp:include page="info.include.jsp" flush="true">
        <jsp:param name="name" value="SuperMan" />
        </jsp:include>
</center>
</body>
</html>
```



其中被包括的页面是 info.include.jsp，该页面的内容是：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<%@page import="java.io.BufferedOutputStream"%><html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset= gb2312">
<title>info of include</title>
</head>
<body>
This page is included page.
<br/>
<%
String name=request.getParameter("name");
%>
您好：<%=name %>
</body>
</html>
```

在浏览器中运行 jsp_include_Demo.jsp 的结果如图 7-43 所示。



图 7-43　运行 jsp_include_Demo.jsp 的结果

（2）<jsp:useBean>动作

<jsp:useBean>动作用来装载一个将在 JSP 页面中使用的 JavaBean。在 JSP 页面中，JavaBean 可以封装业务逻辑，使业务处理和页面显示分开进行，有利于代码重用和程序模块化设计。<jsp:useBean>的语法格式如下：

```
<jsp:useBean id="beanInstanceName" scope="page|request|session|application" />
```

其中各个属性的含义如下。

- beanInstanceName：创建 JavaBean 的实例对象名称，对 JavaBean 进行引用。设置 JavaBean 实例 id，用来区分不同的 JavaBean。如果这个 JavaBean 已经创建，那么这个 id 值必须和原来的保持一致。
- scope="page|request|session|application"：设置 JavaBean 存在的范围，其存在于如下 4 种范围。

- ☑ page：可以在本页面中使用 JavaBean，直到页面发生改变时为止。page 是 scope 的默认值。
- ☑ request：在任何执行相同请求的 JSP 页面中使用 JavaBean，当 request 请求对象发生改变时终止。
- ☑ session：可以在任何使用相同的 session 的 JSP 页面中使用 JavaBean。
- ☑ application：可以在任何使用过相同的 application 的 JSP 页面中使用 JavaBean。

例 7.9　一个使用 Eclipse 建立 JavaBean 的示例。

打开 Eclipse，新建一个 Web/Dynamic Web Project，也可以使用已有的动态网站项目，本例中是 jsp_action。在 jsp_action 项目浏览器上单击鼠标右键，在弹出的快捷菜单中选择 New→Class 命令，新建一个 Java 类，如图 7-44 所示。

在 Package 文本框中输入包名，如“hello”，在 Name 文本框中输入“HelloJavaBean”为类名，其他选择默认即可，如图 7-45 所示。

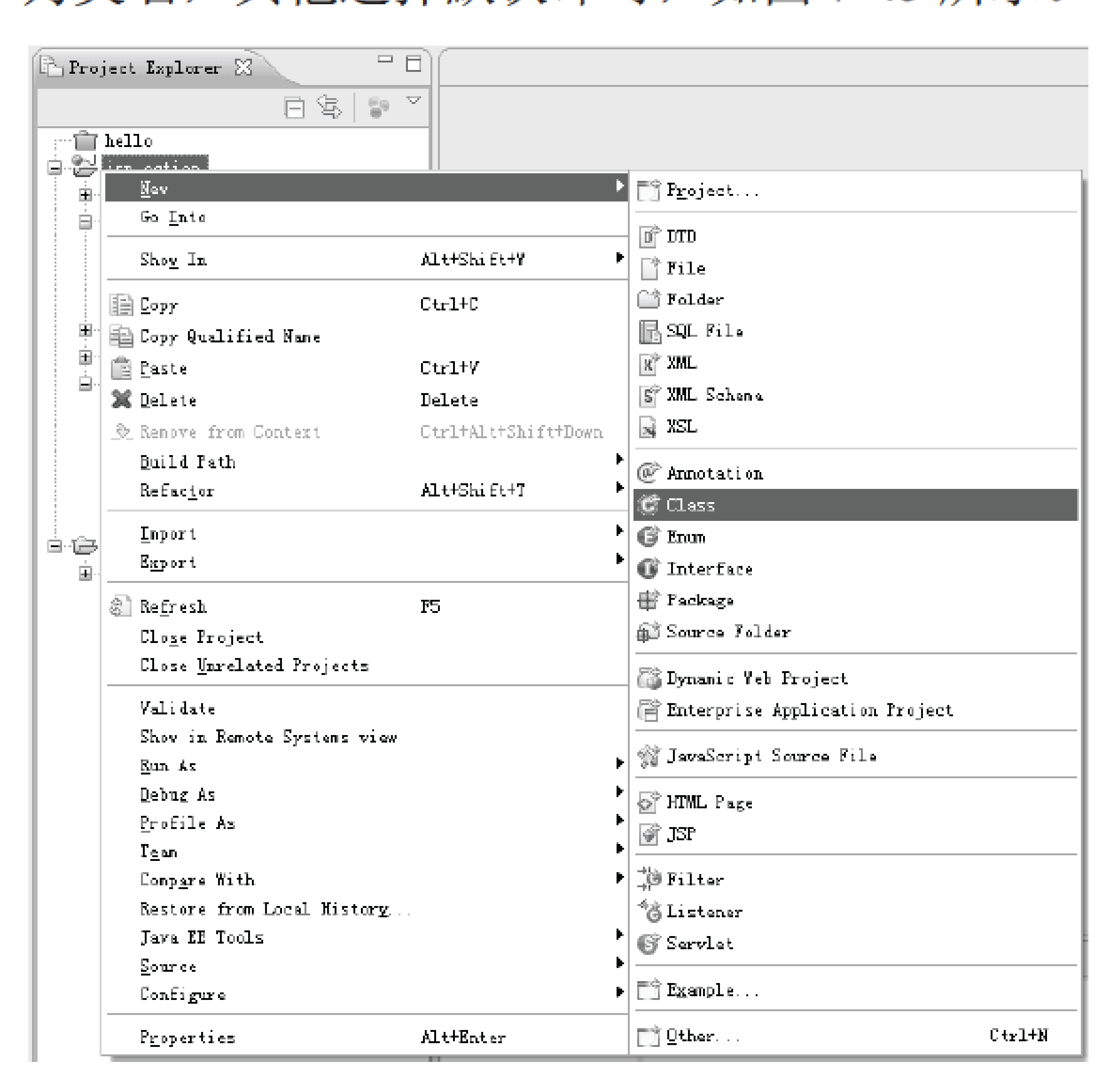


图 7-44　新建一个 Java 类

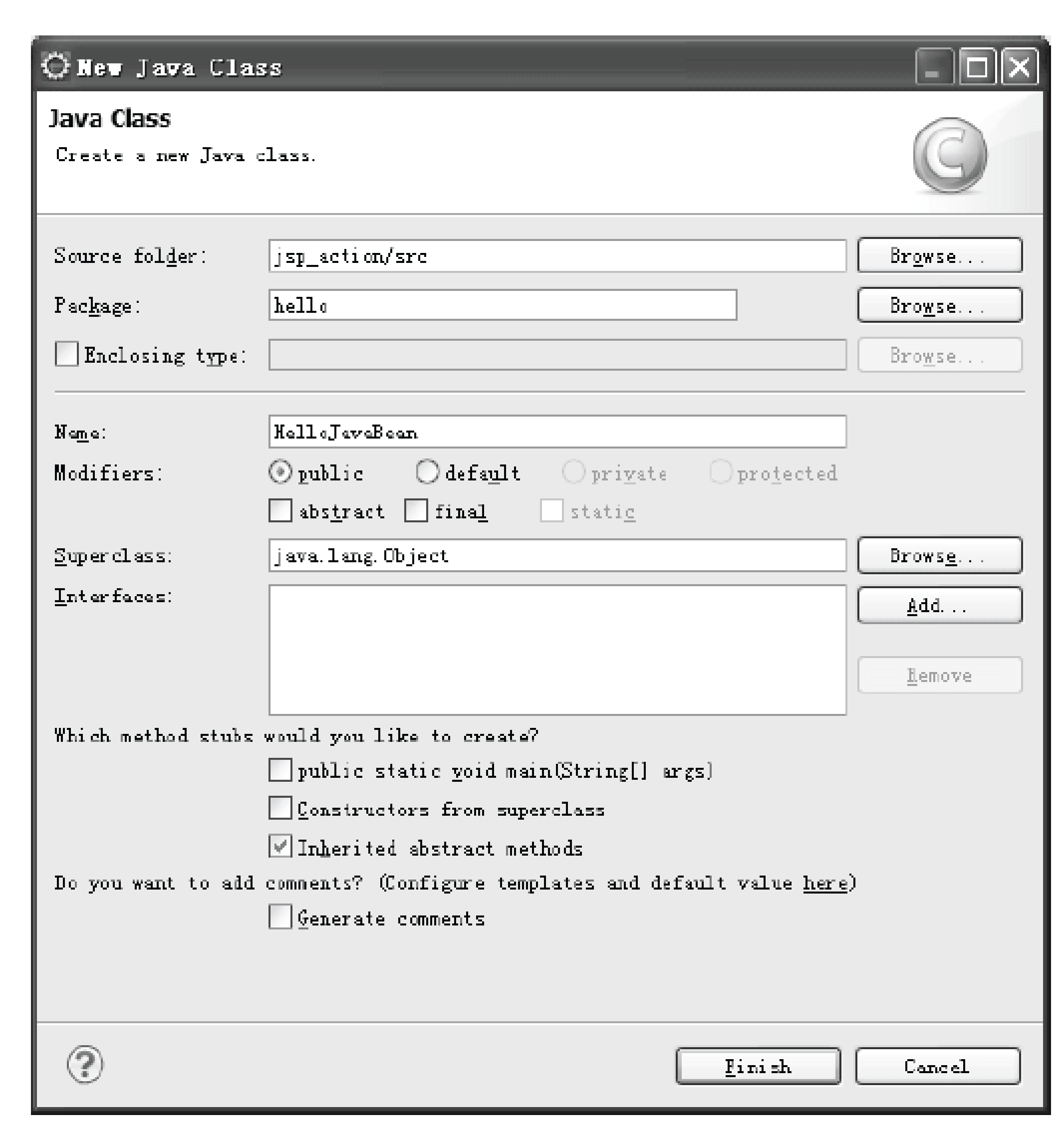


图 7-45　填写 HelloJavaBean 类名

单击 Finish 按钮后，出现 HelloJavaBean.java 的编辑页面，在其中输入以下代码：

```
package hello;
public class HelloJavaBean {
    String str="";
    public String Hello(){
        if (str=="")return "nothing";
        else return str;
    }
    public void SetHello(String hello){
        this.str=hello;
    }
}
```



编辑页面如图 7-46 所示。

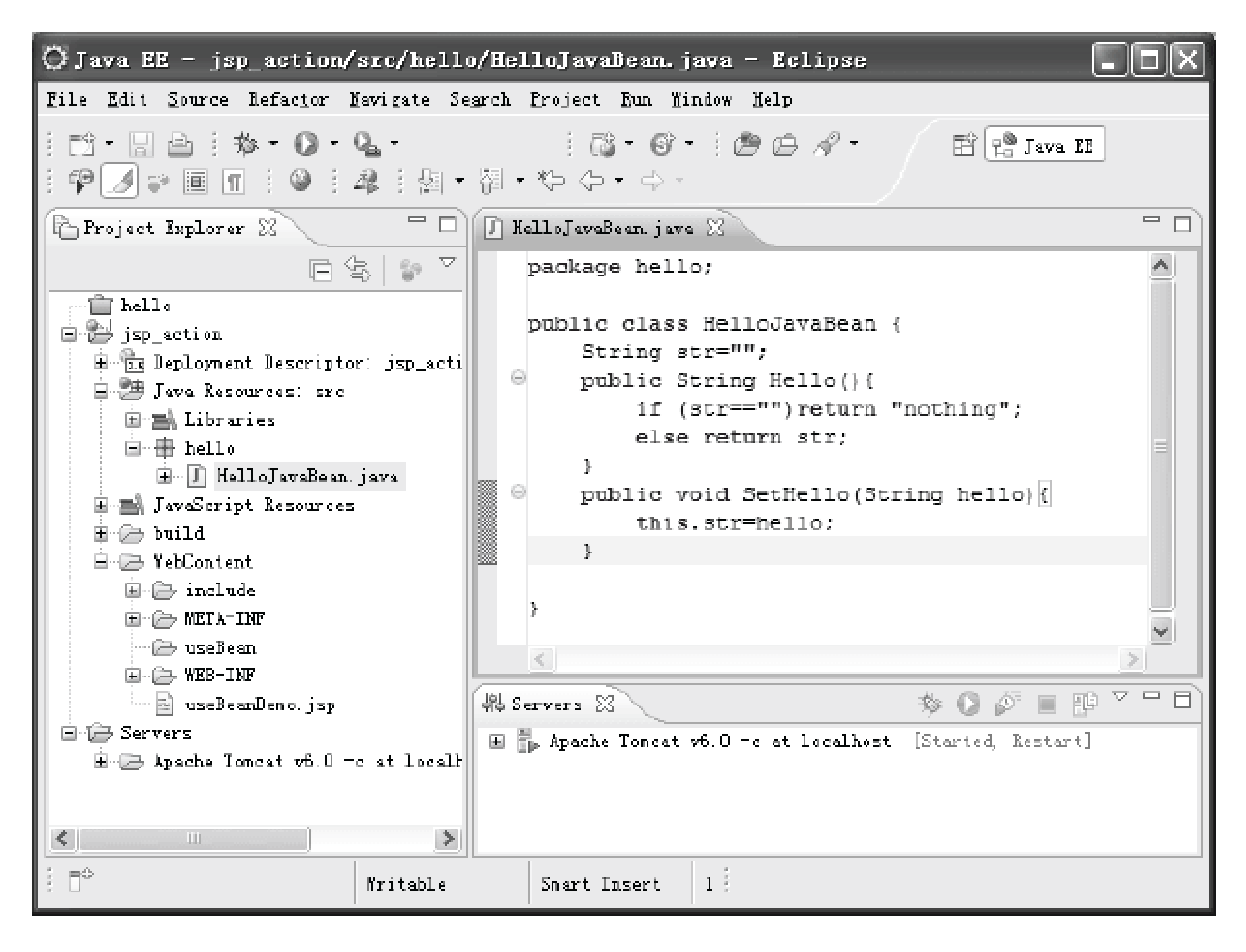


图 7-46　编辑页面

在 WebContent 目录下新建一个 JSP 页面，即在 WebContent 文件夹上单击鼠标右键，在弹出的快捷菜单中选择 New→JSP 命令，然后在 FileName 文本框中输入“useBeanDemo.jsp”，单击 Finish 按钮即可建立一个 jsp 页面，并取名为 useBeanDemo.jsp，其内容如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<jsp:useBean id="SampleJavaBean" scope="page" class="hello.HelloJavaBean" />
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>Insert title here</title>
</head>
<body>
<%// <jsp:useBean id="beanInstanceName" scope="page|request|session|application" /> %>

useBean 的结果 1 是：<%=SampleJavaBean.Hello() %>
<br/>
<%
String str=new String("你好!");
SampleJavaBean.SetHello(str);
%>
useBean 的结果 2 是：<%=SampleJavaBean.Hello() %>
</body>
</html>
```

保存所有页面之后，按 Shift+Alt+X 组合键再按 R 键，即可快速运行本示例，其结果如图 7-47 所示。

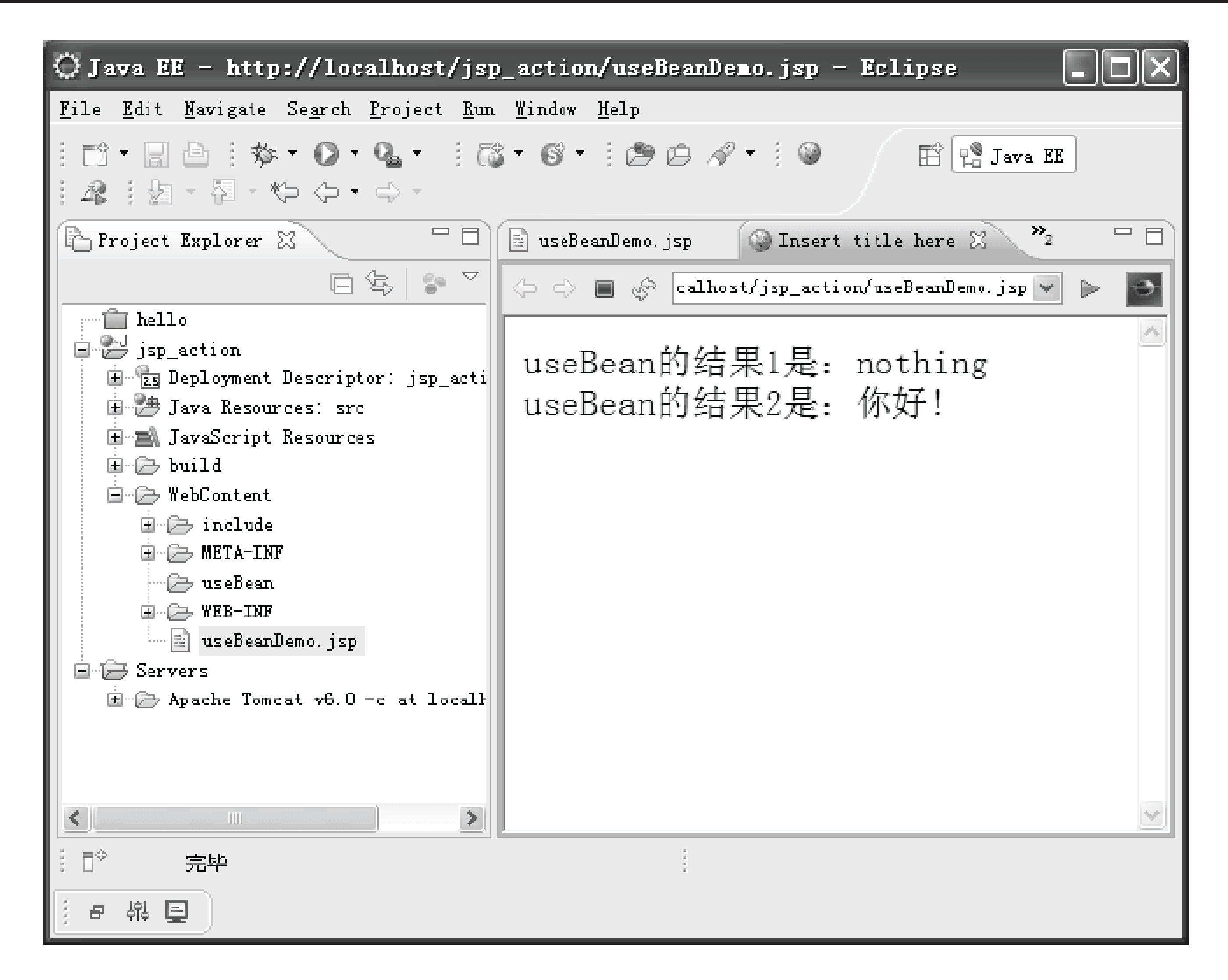


图 7-47　运行结果

（3）<jsp:setProperty>动作

<jsp:setProperty>动作用来设置 JavaBean 的属性值。该动作的使用前提是：必须在<jsp:useBean>动作后至少获得了一个 JavaBean 对象的实例。其语法格式有如下 4 种格式：

```
<jsp:setProperty name="BeanInstanceName"
property="*"|
property="propertyName"|
property="propertyName" param="paramName"|
property="propertyName" value="beanValue"
/>
```

其中，BeanInstanceName 表示 JSP 使用<jsp:useBean>动作引用的 JavaBean 的实例对象名称；propertyName 表示 JavaBean 属性名；paramName 表示 Request 对象的参数名；beanValue 表示 JavaBean 的属性值。

在 JSP 程序中使用<jsp:setProperty>动作时应注意如下事项：

- 在使用属性“property="*"”时，JavaBean 中属性的名字必须和 Request 对象的参数一致。为了匹配 JavaBean 中的属性值，JSP 页面的输入值在必要时要通过数据类型转换。
- 如果 Request 对象的参数中存在控制，则对应的 JavaBean 将不设任何值。如果 JavaBean 中有的属性没有与之相对应的 Request 对象参数，则其属性也不会设定任何值。
- 当 Request 对象的参数和 JavaBean 中的参数名相同时，只需要指定属性 property 即可，否则必须同时指定 property 属性和 param 属性。
- 用于设定 JavaBean 属性值必须和要设定的属性值的数据类型一致，否则必须进行数据类型转换。
- 不能在同一个<jsp:setProperty>中使用参数 param 和 value。



（4）<jsp:getProperty>动作

<jsp:getProperty>动作用来获取 JavaBean 的属性值，其语法格式为：

<jsp:getProperty name="BeanInstanceName" property="propertyName" />

其中，BeanInstanceName 表示 JSP 使用<jsp:useBean>动作引用的 JavaBean 的实例对象名称；propertyName 表示 JavaBean 属性名。通常<jsp:getProperty>要和<jsp:setProperty>动作一起配套使用。

在 JSP 程序中使用<jsp: getProperty>动作时应注意的事项为：不能使用<jsp:getProperty>动作来检索一个已经被索引的属性；使用<jsp:getProperty>动作时，必须在它之前使用<jsp:useBean>动作；<jsp:getProperty>动作不能和 EnterPriseBean 一起使用。

例 7.10　<jsp:getProperty>和<jsp:setProperty>动作应用示例。

在 useBeanDemo.jsp 页面中 body 结束标签前加上下面代码：

```
<br/>setProperty:<br/>
<%!String helloStr; %>
<%
helloStr=new String("Hello_By_setProperty");
%>
<jsp:setProperty name="SampleJavaBean" property="id" value="<%=helloStr %>"/>
getProperty:<br/>
<jsp:getProperty name="SampleJavaBean" property="id" />
```

在 HelloJavaBean.java 中，替换为下面的代码：

```
package hello;
public class HelloJavaBean {
        private String str=new String("");
        private String name;
        public String Hello(){
              if (str=="")
                    return "nothing";
              else return str;
        }
        public void SetHello(String hello){
              this.str=hello;
        }
        public void setName(String name){
              this.name=name;
        }
        public String getName(){
              return this.name;
        }
}
```

运行 useBeanDemo.jsp 页面，结果如图 7-48 所示。

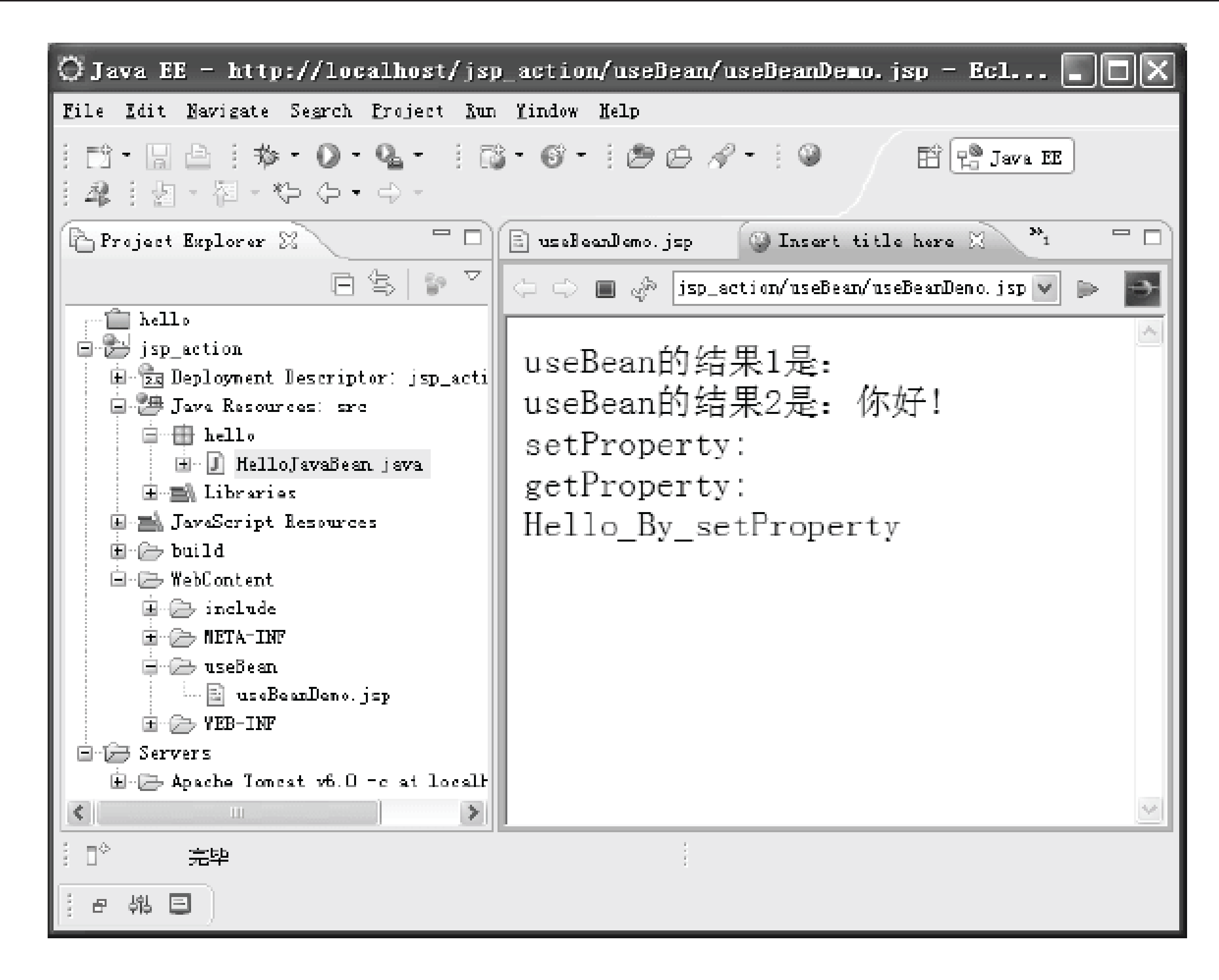


图 7-48　运行 useBeanDemo.jsp 页面

需要注意的是，在 Java 源代码中必须有对类的变量的 set 和 get 方法，而且 set 和 get 后面必须是变量名，这样 getProperty 才能够成功获得需要的变量的 get 方法。

（5）<jsp:forward>动作

<jsp:forward>动作用来将当前的请求转向另外的 JSP 页面、静态文件或一个 Servlet。其语法格式为：

<jsp:forward page="relativeURL|<%=expression%>"/>

对 JSP 页面传参语法格式如下：

<jsp:forward page="relativeURL|<%=expression%>">

<jsp:param name="paramName" value="paramValue" />

</jsp:forward>

其中，参数“page="relativeURL|<%=expression%>"”为一组相对路径，它可以是一个字符串，也可以是一个表达式，paramName 表示参数名；paramValue 表示参数值。

例 7.11　<jsp:forward>动作应用示例。

首先这是两个页面，第 1 个是 getInfo.jsp，内容如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>getInfo page</title>
</head>
<body>
<%!String name; %>
<%
String s=new String();
s=request.getParameter("name");
```



```
if(s==null){
        name=new String("游客");
}else{
        name=s;
}
%>
        <center>
                你好：
<%
out.println(name);
%>
        </center>
</body>
</html>
```

第 2 个页面是 forward.jsp，内容如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
      pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>forward page</title>
</head>
<body>
<jsp:forward page="getInfo.jsp">
 <jsp:param name="name" value="I am SuperMan" />
</jsp:forward>
</body>
</html>
```

运行 forward.jsp 页面，结果如图 7-49 所示。

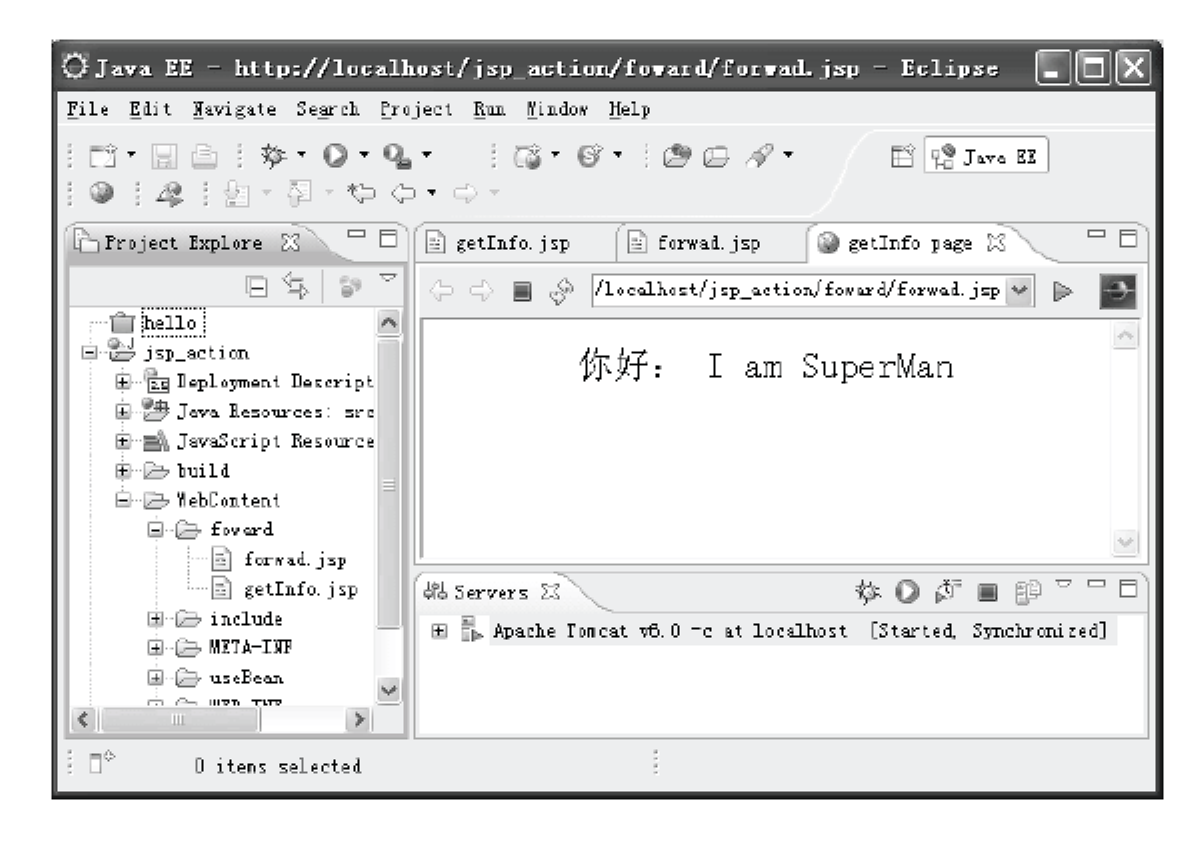


图 7-49　运行 forward.jsp 页面

注意：url 仍然是 forward.jsp，但标题已经是 getInfo.jsp 页面的标题 getInfo page 了。

（6）<jsp:plugin>动作

<jsp:plugin>动作用来在客户端的浏览器中显示一个对象，使 JSP 文件中嵌入客户端运行的 Java 程序通常为 applet 或 Bean。其语法格式如下：

```
<jsp:plugin
type=bean|applet
code="classFileName"
codebase="classFileDirectoryName"
[name="instanceName"]
[archive="archiveList"]
[align="alignment"]
[height="height"]
[width="width"]
[hspace="hspace"]
[vspace="vspace"]
[jreversion="jreversionVersion"]
[title="title"]
[nspluginurl="url"]
[iepluginurl="url"]
>
[<jsp:params>
    <jsp:param name="paramName" value="paramValue" />
</jsp:params>]
[<jsp:fallback>text</jsp:fallback>]
</jsp:plugin>
```

其中各个属性含义如下。

- type=bean|applet：设置被执行对象的类型，该参数的值必须为 bean 或 applet。
- code="classFileName"：设置将要执行的 Java 类文件的名称，这个文件必须以 class 为扩展名，并且该文件必须存在于 codebase 目录中。
- codebase="classFileDirectoryName"：设置 Java 类文件的路径。如果未指定，则默认使用当前 JSP 页面的目录。
- [name="instanceName"]：实例的名字。
- [archive="archiveList"]：设置用于 codebase 指定目录下文件的路径名。
- [align="alignment"]：设置位置，排列方式有 botton、top、middle、left 和 right。
- [height="height"]：设置所要显示的高度。
- [width="width"]：设置所要显示的宽度。
- [hspace="hspace"]：设置屏幕左右要留下的空间。



- [vspace="vspace"]：设置屏幕上下要留下的空间。
- [jreversion="jreversionVersion"]：设置运行环境所需要的 Java Runtime Environment（JRE）的版本号。
- [title="title"]：设置所要显示的名称。
- [nspluginurl="url"]：设置 Netscape Navigator 用户能够使用的 JRE 的下载地址。
- [iepluginurl="url"]：设置 IE 用户能够使用的 JRE 下载地址。
- <jsp:params>：设置需要向 bean 或 applet 传入的参数。
- <jsp:fallback>：设置 Java 插件 bean 或 applet 不能启动时显示的提示信息。

例 7.12　<jsp:plugin>动作应用示例。

首先新建立一个 Java 文件 HelloWorld.java，内容如下：

```
import java.applet.Applet;
import java.awt.Graphics;
public class HelloWorld extends Applet
{
    String name;
    public void init()
    {
     name = getParameter("name");
    }
    public void paint(Graphics g)
    {
        g.drawString("This demo show jsp:plugin usage,the " + name + " is <br> a parameter!", 40, 25);
    }
}
```

在 cmd 下编译这个文件，运行 javac HelloWorld.java，生成一个 HelloWorld.class 文件，将 HelloWorld.class 文件复制到 Eclipse 的项目下。然后在 Eclipse 的项目中添加一个 jsp 页面 plugin.jsp，内容如下：

```
<!-- PLUGIN.jsp -->
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<HTML>
<HEAD>
    <meta http-equiv="Content-Type" content="text/html; charset=GB2312">
    <TITLE>用&lt; jsp:plugin &gt;加载 Applet</TITLE>
</HEAD>
<BODY>
    <CENTER>
    <FONT SIZE="5" COLOR="blue">用&lt; jsp:plugin &gt;加载 Applet</FONT>
    </CENTER>
```

```
<BR/>
<BR/>
    <CENTER>
        <!-- 用 plugin 加载 applet -->
        <jsp:plugin type="applet" code="HelloWorld" codebase="." height="40" width="500" >
        <jsp:params>
            <jsp:param name="name" value="paramValue"/>
        </jsp:params>
        <jsp:fallback>无法加载 Applet</jsp:fallback>
    </jsp:plugin>
    </CENTER>
</BODY>
</HTML>
```

确定这个 jsp 文件和 HelloWorld.class 文件在同一目录下，然后在 Eclipse 中运行这个文件，在 Eclipse 中的浏览器中并没有显示任何信息，因为 Eclipse 的浏览器并不具备过多浏览器的功能。在 Firefox 和 IE 下同时进行测试，在 Firefox 下的页面如图 7-50 所示，在 IE 下的页面如图 7-51 所示。



图 7-50　Firefox 下的页面



图 7-51　IE 下的页面

由于笔者的计算机上已经安装了 Java 的 JDK（自然包括 JRE），所以可以正常显示，如果没有安装 JRE 则会显示错误提示信息。

7.6　JSP 常用的内置对象

7.6.1　内置对象的概述

JSP 内置对象是不需要声明而直接可以在 JSP 网页中使用的对象，这些内置对象由容器实现和管理，大大简化了页面的开发。所有的内置对象只有对 Scriptlet 或者表达式有效，在 JSP 声明中不可用，因为它们无须声明。

在 JSP 2.0 规范中共定义了 9 种内置对象，分别为 request（请求对象）、response（应答



对象)、pageContext（页面上下对象)、session（会话对象)、application（全局应用程序对象)、out（输出对象）、config（配置对象）、page（页面对象）和 exception（页面以外对象），其中，request、response、session 和 application 最为常见。

从本质上讲，JSP 的这些内置对象都是由特定的 Java 类所产生的，在服务器运行时根据情况自动生成。表 7-11 给出了内置对象与 Java 类的对应关系。

表 7-11　内置对象与 Java 类的对应关系

对 象 名 称	所 属 类 型	有 效 范 围
request	javaax.servlet.ServletRequest	request
response	javaax.servlet.ServletResponse	page
pageContext	javaax.servlet.jsp.PageContext	page
session	javaax.servlet.http.HttpSession	session
application	javaax.servlet.ServletContext	application
out	javaax.servlet.jsp.JspWriter	page
config	javaax.servlet.ServletConfig	page
page	java.lang.Object	page
exception	java.lang.Throwabie	page

7.6.2　处理客户请求信息对象 request

request 对象是从客户端向服务器发出请求的，包括用户提交的信息以及客户端的一些信息。这些请求信息包括请求的来源、表头、Cookies 及请求相关的参数值等。来自客户端的请求信息经 Servlet 容器处理后，由 request 对象进行封装，它作为 jspService()方法的一个参数由容器传递给 JSP 页面。request 对象的主要方法如表 7-12 所示。

表 7-12　request 对象的主要方法

方　　法	功　　能
setAttribute(String name,Object objt)	设置名称为 name 的 request 参数的值
getAttribute(String name)	返回由 name 指定的属性值，如果不存在则返回空
getMethod()	获得客户向服务器传输数据的方式：get、post 等
getRequestURI()	获得发出请求字符串的客户端地址
getAttributeNames()	返回 request 对象所有属性的名字集合
getHeader(String name)	获得 HTTP 协议定义的文件头信息
getParameter(String name)	获得客户传送给服务器的 name 参数的值
getParameterValues(String name)	获得指定参数 name 的所有值
getProtocol()	获得客户端向服务器传送数据所依据的协议名称
getRemoteAddr()	取得客户端的 IP 地址
getRemoteHost()	获得客户端主机名
getRemoteUser()	获得客户端用户名

续表

方　　法	功　　能
getServerName()	获得服务器的名字
getServerPort()	获得服务器的端口号
getContentLength()	获得请求实体数据的大小
getContentType()	获得 MIME 类型
getQueryString()	取得客户以 get 方法向服务器传送的查询字符串
getCookies()	返回客户端的所有 Cookie 对象，一个 Cookie 数组
getSession(Boolean create)	返回和请求相关的 session
getServletPath()	获得客户端所请求脚本文件的相对路径

例 7.13　访问请求参数示例。

在 login.jsp 页面中通过表单向 login_detail.jsp 页面提供数据，在 login_deal.jsp 页面后提交的数据并输出。

（1）编写 login.jsp 页面，在该页面中添加相关的表单即表单元素，关键代码如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>login jsp page</title>
</head>
<body>
 <form id="form"   name="form"   method="post" action="login_detail.jsp">
   用户名:
   <input name="username" type="text" id="username"/><br/>
   密  码:
   <input   name="pwd"    type="text"   id="pwd"/><br/>
   <input type="submit" name="submit" value="提交"/>
   <input type="reset" name="submit2"   value="重设"/>
 </form>

</body>
</html>
```

在 Eclipse 下的运行结果如图 7-52 所示。

（2）编写 login_detail.jsp 页面，该页面中获取提交的数据，关键代码如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
    <%
  String username=request.getParameter("username");
```



```
    String pwd=request.getParameter("pwd");
    out.println("用户名为："+username);
    out.println("密码为："+pwd);
%>
```

图 7-52　login.jsp 页面的运行结果

运行结果如图 7-53 所示。

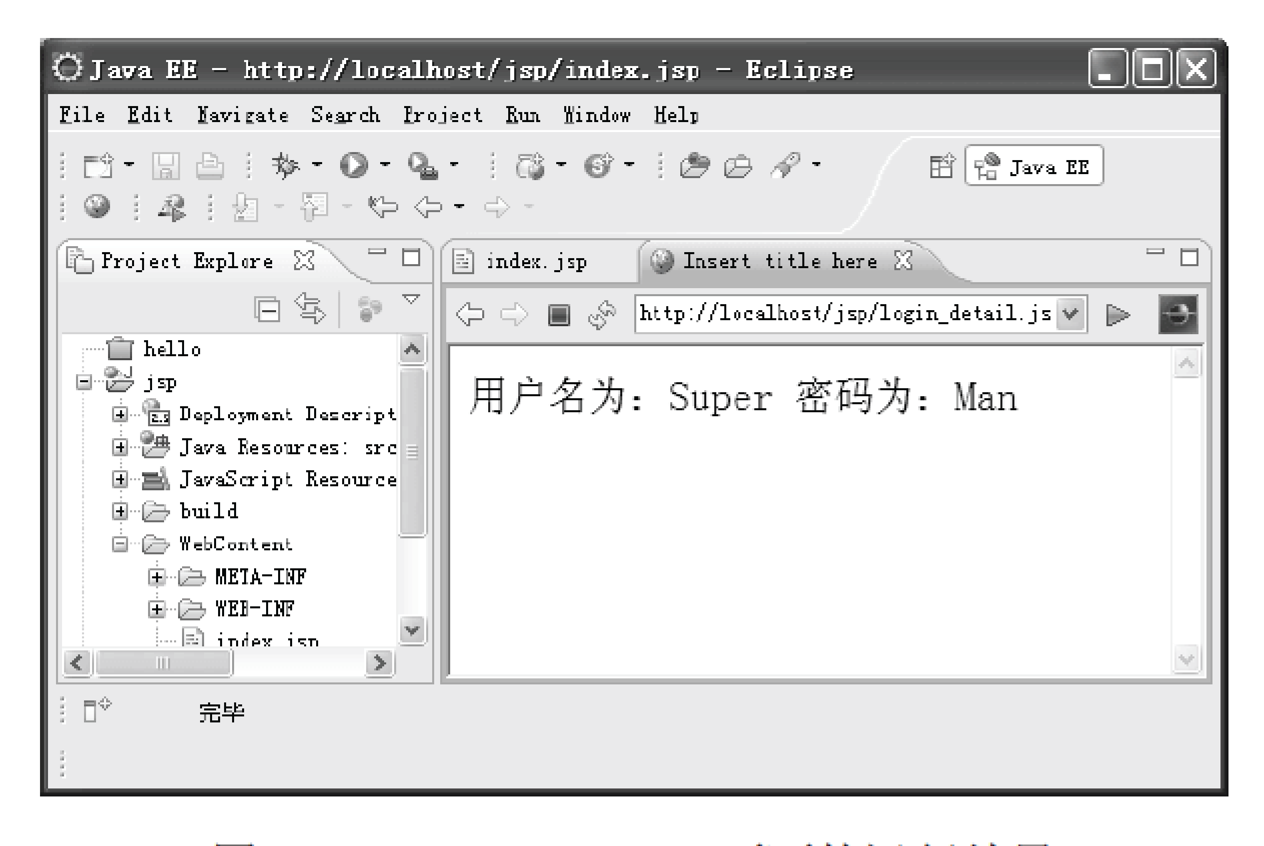


图 7-53　login_detail.jsp 页面的运行结果

7.6.3　控制服务器相应信息对象 response

response 对象和 request 对象相对应，用于响应客户请求，向客户端发送数据。response 对象与该对象提供的方法成员大部分被用来设置服务器端响应给客户端网页的状态，如编码方式、响应表头、错误信息等。response 对象提供的方法如表 7-13 所示。

表 7-13　response 对象提供的方法

分　类	方　法	说　明
设置响应信息	getcharacterEncoding	返回文件内容编码的方式
	setContentType	设置网页的文件格式与编码方式
	sendError	定义输出客户端的错误代码以及信息
	setStatus	用来设置一个回应的 HTTP 状态码

续表

分　类	方　法	说　明
响应表头信息	ContainsHeader(String name)	返回布尔值表示名称为 name 的表头是否存在
	SetDateHeader (String name,long date)	设置名称为 name 的响应表头，数据类型为长整数 long，内容为 date，其值为自格林威治时间 1970 年 1 月 1 日开始算起到要设置时间的毫秒数
	setHeader(String name,String value)	设置名称为 name 的响应表头，其内容为 value，数据类型为字符串 String
	setIntHeader(String name,int value)	设置名称为 name 的响应表头，其内容为 value，数据类型为整数 int
缓冲区处理	flushBuffer()	清空缓冲区
	getBufferSize()	取得缓冲区大小
	setBufferSize()	设置缓冲区大小
	IsCommitted()	表示数据是否清除完毕，写入浏览器
	Reset()	清除缓冲区内容
网页定向与重设	sendRediret("pageUrl")	重设与定向网页到指定的 URL

1．设置响应信息

当网页传送至客户端浏览器时，用户可以设置所要响应的信息和网页状态，利用 response 对象所提供的方法来完成，这些方法包括 getcharacterEncoding()、setContentType 及 sendError()等。

（1）getcharacterEncoding 方法

getcharacterEncoding 方法会返回文件内容编码的方式，其语法格式如下：

response.getcharacterEncoding()

默认情况下返回的值是 8859--1，但若是该网页使用 page 指令或者 setContentType()方法设置了网页的编码方式，则返回的值是 GB2312。

（2）setContentType 方法

setContentType 方法是用来设置网页的文件与编码方式的，其与 page 指令的 contentType 属性功能是相同的。如下面这两行代码，其运行结果均完全相同：

```
<%@page contentType =     "text/html;charset=GB2312"%>
response.setContentType("text/html;charset=GB2312");
```

（3）sendError 方法

sendError 方法可以自定义一个错误的代码，其语法格式如下：

Response. SendError(错误代码,文字信息)

对于用户的不当操作，程序开发人员可以自行设置容易理解的错误信息，利用 sendError 将其送出，以提供比较出色的交互功能。

（4）setStatus 方法

setStatus 方法可用来设置一个相应的 HTTP 状态码，根据这个状态码，客户端便会出现相应的信息。在一般情况下，若打开的网页正确无误，则会自动传送一个 SC--OK（整数值





为 200）的状态码。

例 7.14　使用 setStatus 方法来设置回应的状态码，查看客户端的浏览器会出现怎样的信息。

usingsetStatus.jsp 页面的代码如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>setStatus 方法</title>
</head>
<body>
     <% response.setStatus(403); %>
    Have setStatus 403
</body>
</html>
```

运行结果如图 7-54 所示。

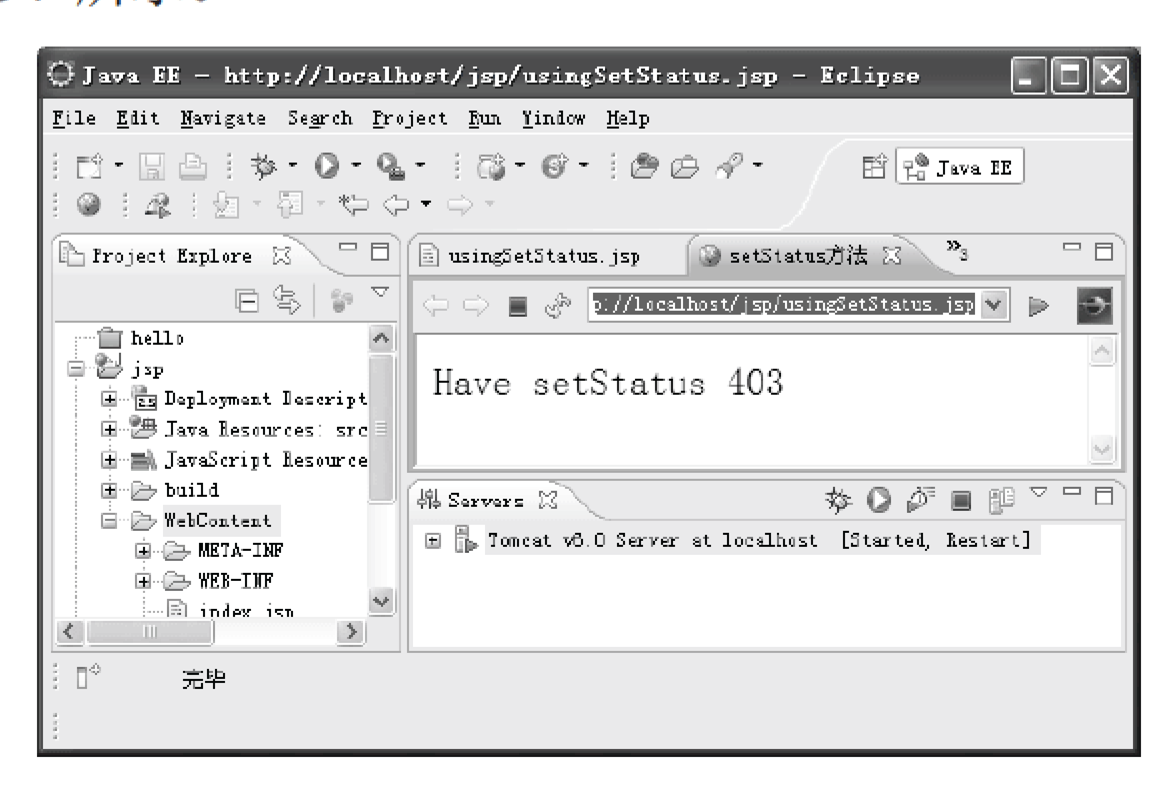


图 7-54　运行结果

2. 响应表头信息

前面在说明 HTTP 时曾提到过，表头信息可分为请求表头和响应表头，response 对象提供了响应表头信息处理有关的方法，在表 7-13 中已分类作了简要的说明。由于 HTTP 各种响应表头中可能包括不同的数据类型，因此表 7-13 中包括了用来设置不同表头数据类型的方法，不过最常使用的还是 setHeader()方法，例如：

```
<%
response.setHeader("Connection", "keep--alive");        //设置 Connection 表头的内容
response.setDateHeader ("Expires",10000*10000);         //设置 Expires 表头的内容
response.setDateHeader("Retry--After",10000);
%>
```



3．操作缓冲区配置

缓冲可以更加有效地在服务器与客户之间传输内容。HttpServletResponse 对象为支持 jspWriter 对象而启用了缓冲区配置。

response 对象中有一类方法成员专门用来处理缓冲区的有关操作。此外，response 还提供了设置缓冲大小的 setBufferSize 缓冲区直接操作。

例 7.15　用 response 进行缓冲区的操作。

```
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
 <html>
   <head><title>缓冲区</title></head>
   <meta http-equiv="Content-Type" content="text/html; charset=GB2312">
       <body>
目前缓冲区大小：<%response. getBufferSize();%><br/>
重设缓冲区大小：16384 <% response.setBufferSize(16384); %><br/>
重设后缓冲区大小：<%=response. getBufferSize()%><br/>
       </body>
 </html>
```

运行结果如图 7-55 所示。

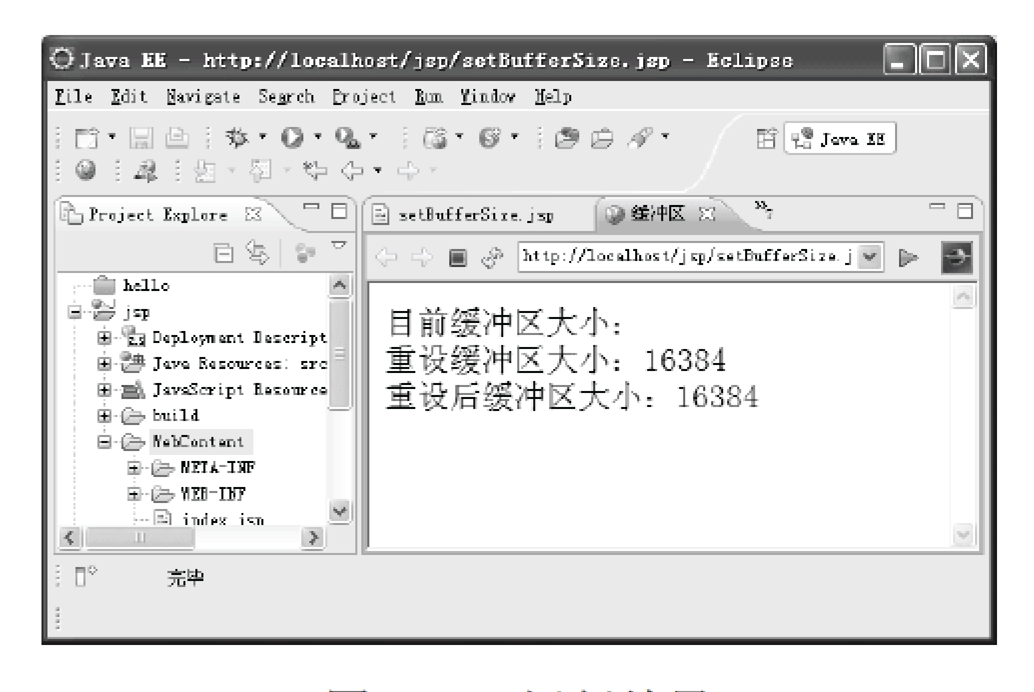


图 7-55　运行结果

4．重新定向网页

重新定向网页包括自动更新网页和定向网页两种情况。在 JSP 中，可以使用 response 对象中的 sendRedirect 方法将客户请求重新定向新网页。其语法格式如下：

response. sendRedirect("定向网页的相对和绝对地址")

例 7.16　使用 sendRedirect 方法重新定向网页。

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
```





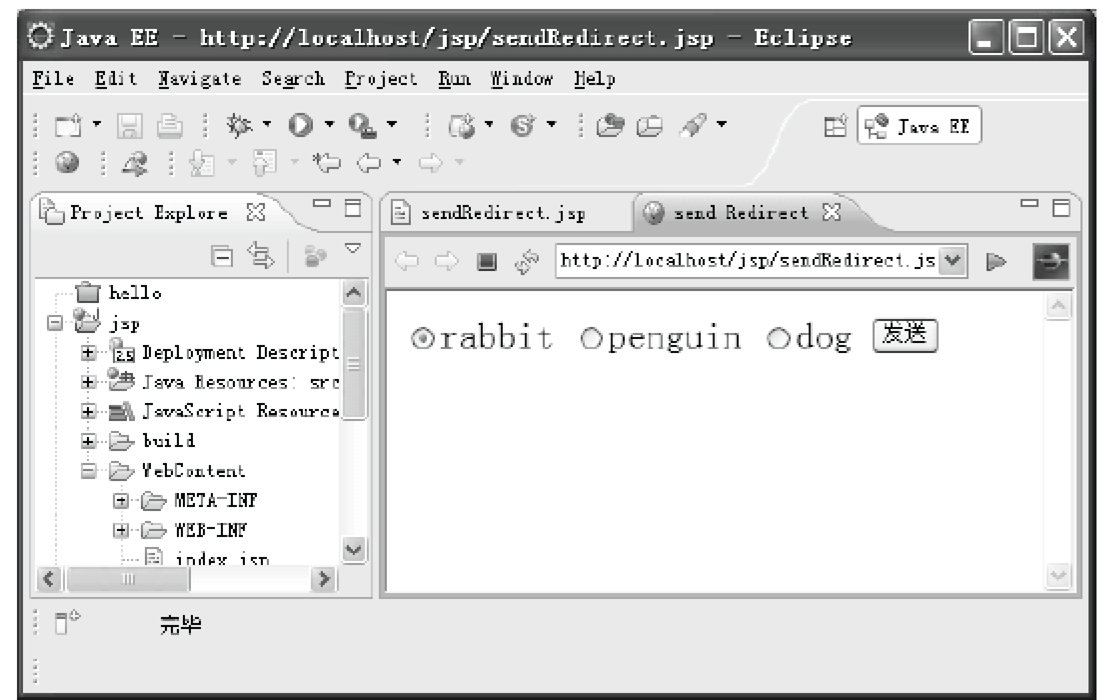
```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>send Redirect</title>
</head>
<body>
<form   action= "usingSendDirect.jsp" method ="post">
        <input type ="radio" name ="view" value ="rabbit" checked />rabbit
        <input type ="radio" name ="view" value ="penguin" />penguin
            <input type ="radio" name ="view" value ="dog">dog
        <input type ="submit" value ="发送" />
</form>
</body>
</html>
```

其中 usingSendDirect.jsp 的内容如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
      pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>usingSendDirect jsp page</title>
</head>
<body>
view=
<%
String view=request.getParameter("view");
out.println(view);
%>
</body>
</html>
```

sendRedirect.jsp 的运行页面如图 7-56 所示。

单击“发送”按钮之后的页面如图 7-57 所示。



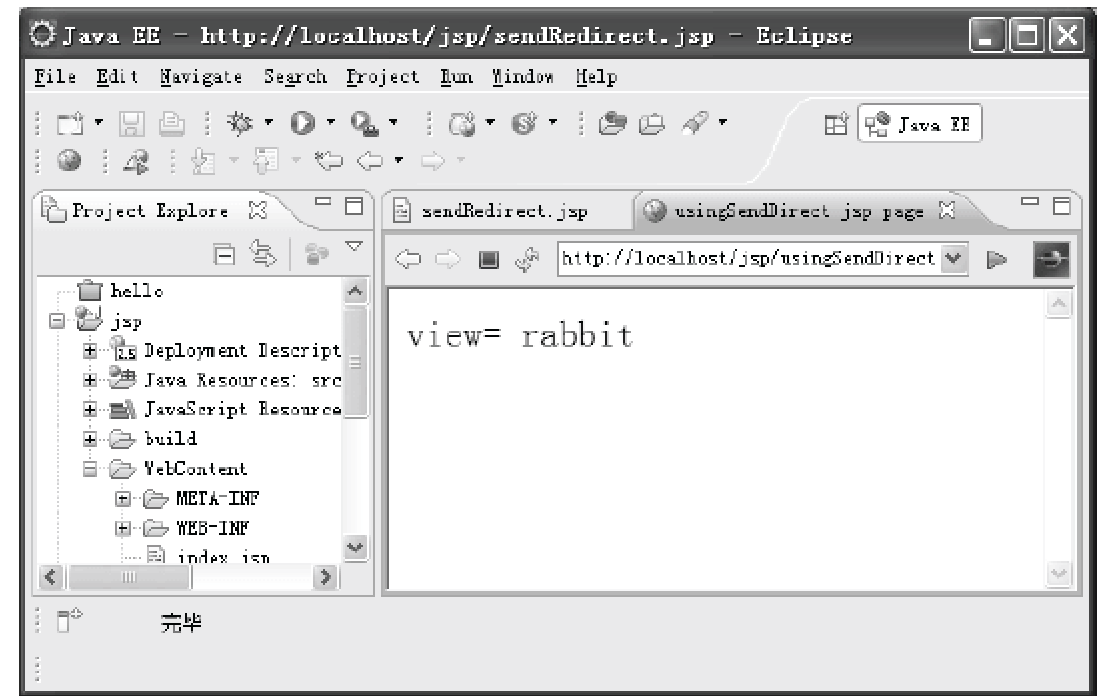




图 7-56　sendRedirect.jsp 页面的运行结果　　　图 7-57　using SendRedirect 页面的运行结果

7.6.4 管理客户会话对象 session

session 对象是由服务器自动产生的，用于保护每个用户信息。每一个 session 都是独立的，且其中数据内容互不相干，每个用户网页所读取的数据也就不同。JSP 页面可以将任何对象作为属性来保存。session 内置对象使用 setAttribute()和 getAttribute()方法创建及获取客户的会话。

当用户首次访问一个 JSP 页面时，JSP 引擎产生一个 session 对象，同时分配一个 String 类型的 ID 号。JSP 引擎将这个 ID 号发送到客户端，存放在 Cookie 中，这样 session 对象和客户之间就建立了一一对应关系。session 对象的常用方法如表 7-14 所示，其中最为常见的方法是 getId()、isNew()、getAttribute()和 setAttribute()。

表 7-14 session 对象的常用方法

方 法 名 称	方 法 描 述
getAttribute(String name)	获得与指定名称 name 相联系的属性值
setAttribute(String name,Object value)	设置指定名称 name 的属性值 value，存储在 session 对象中
removeAttribute(String name)	删除与指定 name 相联系的属性
getAttributeNames()	返回 session 对象中存储的每一个属性对象，其结果为一个枚举类的实例
getCreationTime()	返回 session 被创建的时间，1970.1.1 至今的毫秒数
getId()	返回 session 的标识
getLastAccessedTime()	返回当前 session 对象相关的客户端最后发送请求的时间，最小单位为千分之一秒
Invalidate()	销毁 session 对象，使得和它绑定的对象都失效
isNew()	检查客户端是否参加了会话，判断是否是一个新的客户
getValue(String name)	返回 session 中名为 name 的对象的值，name 不存在则返回 null

例 7.17 使用 session 制作站点计数器。

```
<%@page contentType="text/html;charset=gb2312"%>
<html>
<body>
<%! int number=0;
synchronized void countpeople()
        {
            number++;
        }
%>
<%
 if(session.isNew())
{
```



```
        out.println("A New Session is start...");
        countpeople();
        String str=String.valueOf(number);
        session.setAttribute("count",str);
    }
    %>
<p>您是第<%=(String)session.getAttribute("count")%>个访问本站的人。</p>
</body>
</html>
```

运行此页面的效果如图 7-58 所示（其中，“A New Session is start...”只在第一次显示，刷新后就不会显示，除非产生新会话）。

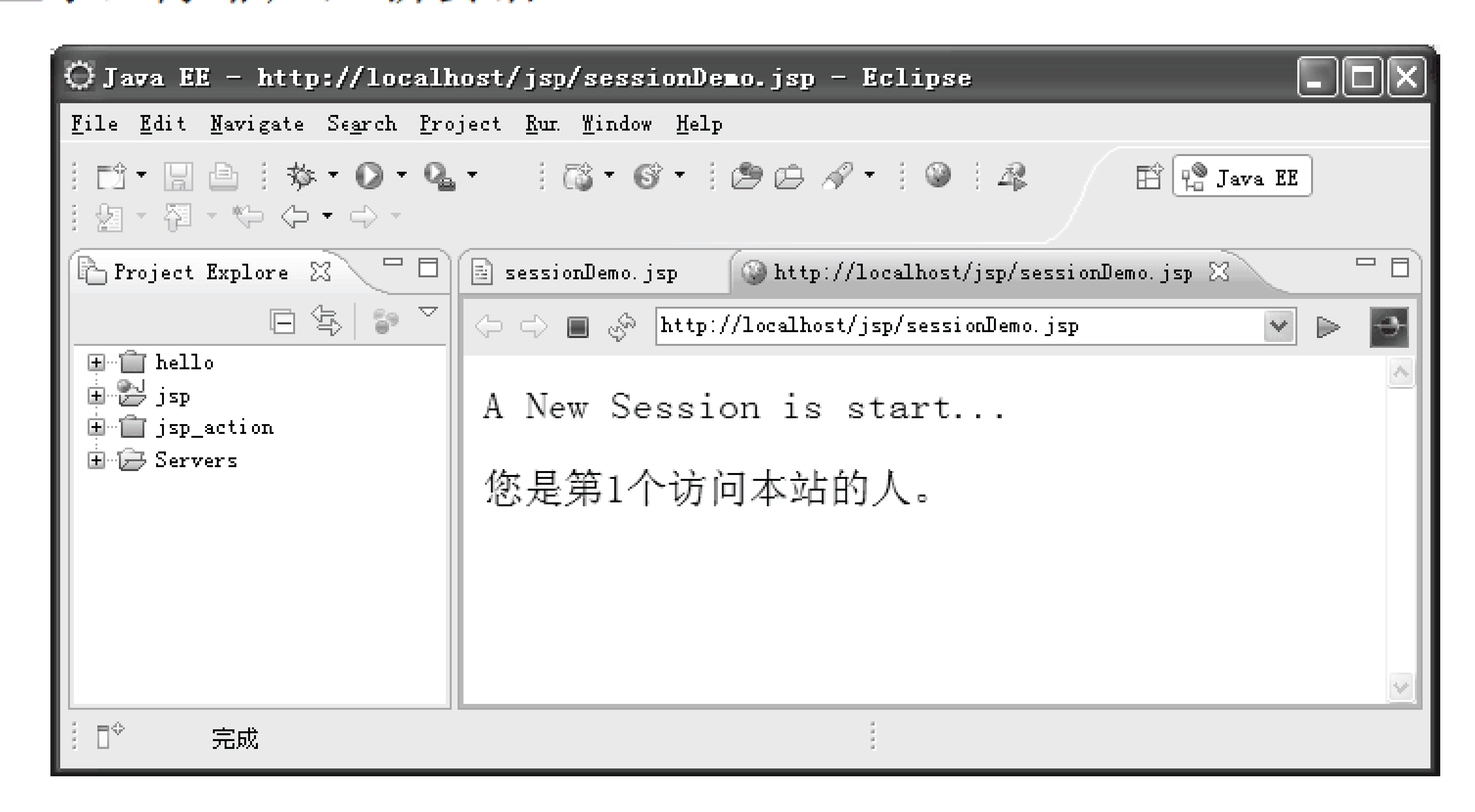


图 7-58　运行效果图

7.6.5　Web 应用程序全局对象 application

application 对象用于保存所有应用程序中的公有数据，服务器启动并且自动创建 application 对象后，只要没有关闭服务器，application 对象将一直存在，所有用户可以共享。

服务器启动后就产生了这个 application 对象，当客户在所访问的网站的各个网页之间浏览时，这个 application 对象都是一个，直到服务器关闭。与 session 不同的是，所有客户的 application 对象都是一个，即所有客户共享这个内置的 application 对象。application 对象的常用方法如表 7-15 所示。

表 7-15　application 对象的常用方法

方　法	功　能
getAttribute(String name)	获得指定名字的 application 对象属性的值
getAttribute(String name,Object obj)	用 object 来初始化某个由 name 指定的属性
getAttributeNames()	返回 application 对象存储的每一个属性的名字
getInitParameter(String name)	返回 application 对象某个属性的初始值
removeAttribute(String name)	删除一个指定的属性
getServerInfo()	返回当前版本 Servlet 编辑器的信息

续表

方　　法	功　　能
getContext(URI)	返回指定 URI 的 serletContext
getMajorVersion()	返回 ServletAPI 的版本
getMimeeType(URI)	返回指定 URI 的文件格式
getRealpath(URI)	返回指定 URL 的实际路径

例 7.18　下面举例说明 application 对象的使用方法。这个例子包含两个网页文件：applicationData.jsp 和 usingApplication.jsp。

下面是 applicationData.jsp 页面的代码：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>设置 application 数据</title>
</head>
<body>
<%
        application.setAttribute ("title","题目");
        application.setAttribute("author","作者");
%>
<a href="usingApplication.jsp">显示设置的 application 数据内容</a>

</body>
</html>
```

下面是 usingApplication.jsp 页面的代码：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>使用 application  数据</title>
</head>
<body>
title=<%=application.getAttribute ("title") %><br/>
author=<%=application.getAttribute("author") %><br/>
</body>
</html>
```



applicationData.jsp 页面的运行结果如图 7-59 所示。

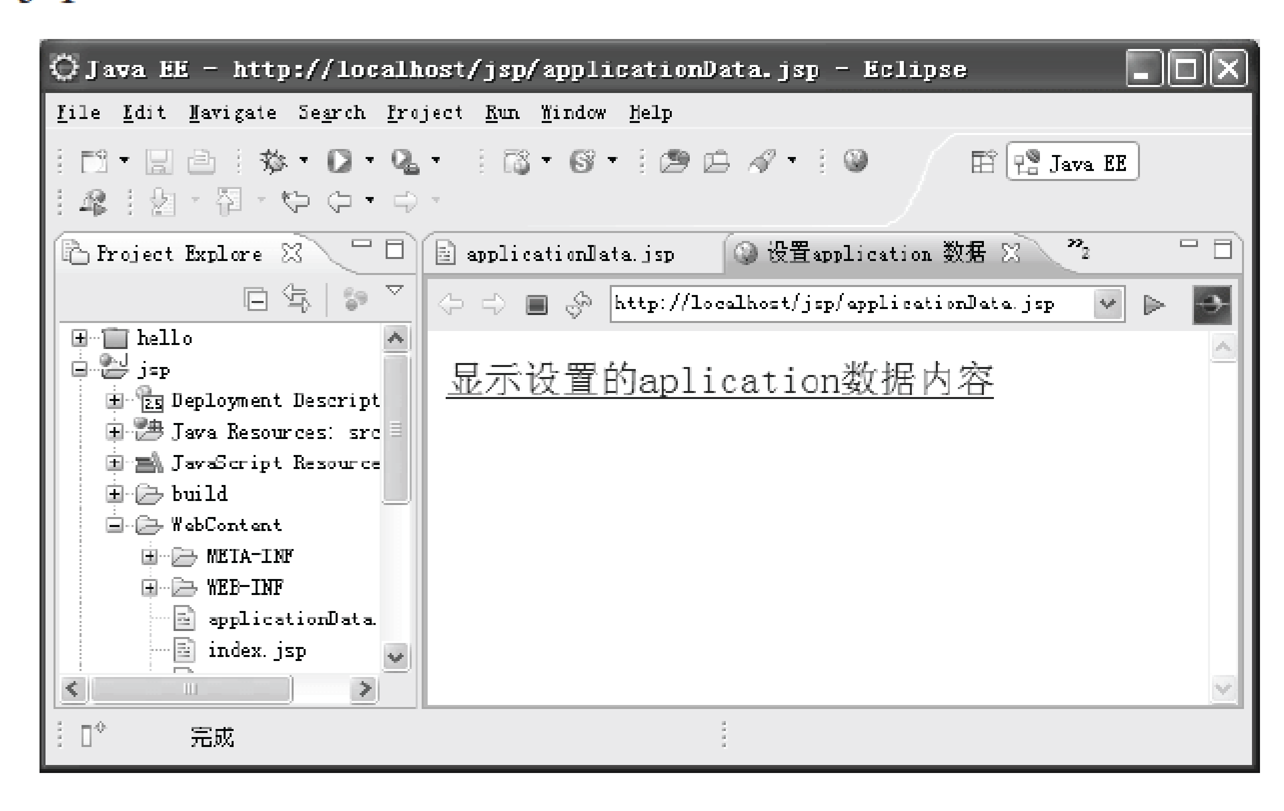


图 7-59　applicationData.jsp 页面的运行结果

单击"显示设置的 application 数据内容"超链接后，跳转到 using Application.jsp 页面，结果如图 7-60 所示。

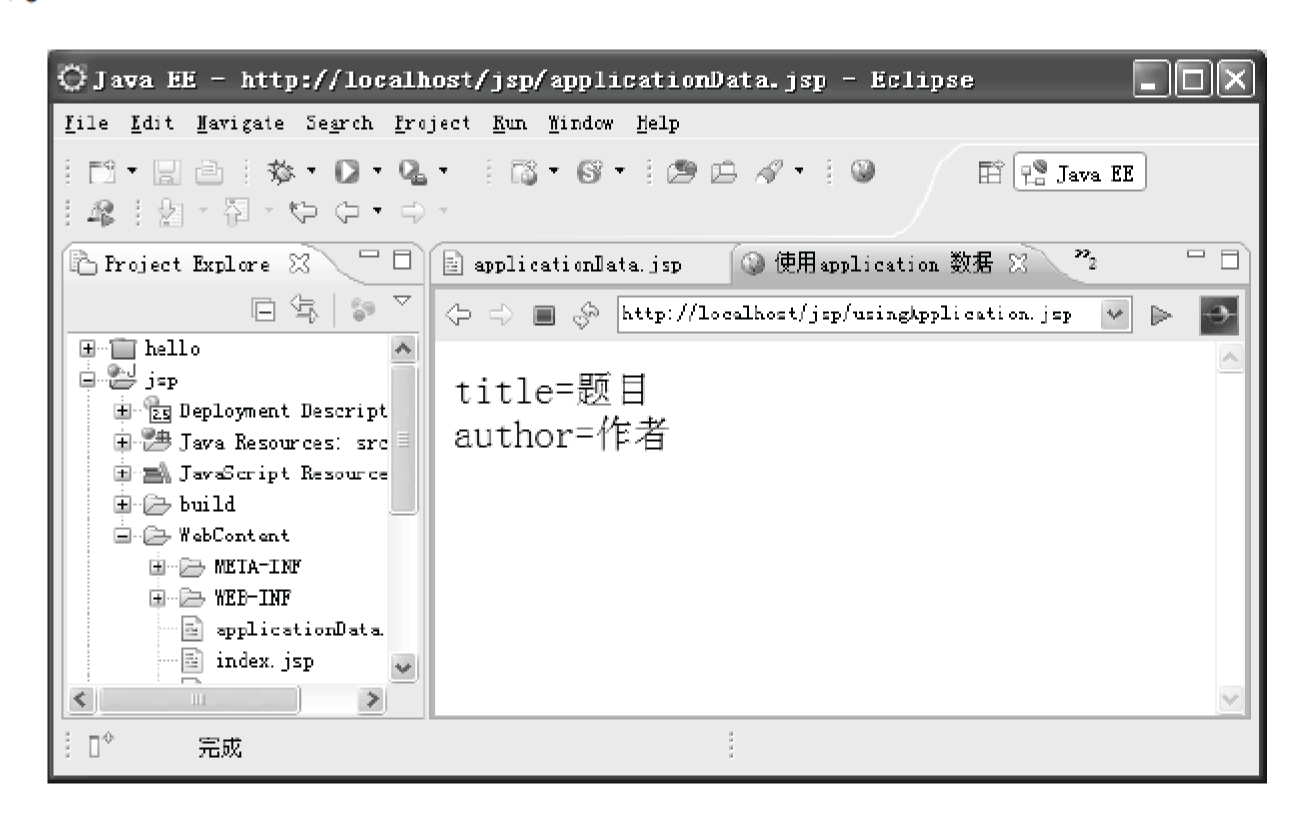


图 7-60　using Application.jsp 页面的运行结果

7.6.6　向客户输出数据对象 out

out 对象主要用来向客户端输出各种数据类型的内容，并且管理应用服务器上的输出缓冲区，缓冲区默认值一般是 8KB，可以通过页面指令 page 来改变其默认值。在使用 out 输出数据时，可以对数据缓冲区进行操作，及时清除缓冲区中的残余数据，为其他的输出让出缓冲空间。

在 JSP 页面中，可以通过 out 对象调用 clear()方法清除缓冲区的内容。out 对象的常用方法如表 7-16 所示。

表 7-16　out 对象的常用方法

方　法	说　明
clear()	清除缓冲区中的数据，若是空的，则会产生 IOException 的异常
clearBuffer()	清除缓冲区中的数据，若是空的，并不会产生 IOException 的异常

续表

方　法	说　明
flush()	直接将目前暂存于缓冲区的数据输出
getBufferSize()	返回缓冲区的大小
getRemaining()	返回缓冲区剩余的空间大小
isAutoFlush()	返回布尔值表示是否自动输出缓冲区中的数据
newLine()	输出换行
print(datatype data)	输出数据类型为 datatype 的数据 data
println(datatype data)	输出数据类型为 datatype 的数据 data，并自动换行

7.6.7　读取 web.xml 配置信息对象 config

config 对象被封装成 javax.servlet.ServletConfigjiekou，它表示 Servlet 的配置，当一个 Servlet 初始化时，容器把某些信息通过此对象传递给这个 Servlet。开发者可以在 web.xml 文件中为应用程序环境中的 Servlet 程序和 JSP 网页提供初始化参数。config 对象的常用方法如表 7-17 所示。

表 7-17　config 对象的常用方法

方　法	说　明
getServletContext()	返回执行者的 Servlet 上下文
getServletName()	返回 Servlet 的名字
getInitParameter()	返回名字为 name 的初始参数的值
getInitParameterName()	返回这个 JSP 所有的初始参数的名字

其中较为常用的是 getInitParameter()和 getInitParameterName()，通过它们可以获得 Servlet 初始化时的参数。

7.6.8　其他 JSP 内建对象

在 JSP 内置对象中，pageContext、page 及 exception 这些对象是不经常使用的，下面进行简单介绍。

1．获取会话范围的 pageContext 对象

pageContext 对象是一个比较特殊的对象，它相当于页面中其他对象功能的最大继承者，使用它可以访问本页中所有其他对象。pageContext 对象被封装成 javax.servlet.pageContext 接口，主要用于管理 JSP 中特殊可见部分中已经命名对象的访问，它的创建都是由容器来完成的。pageContext 对象的常用方法如表 7-18 所示。



表 7-18　pageContext 对象的常用方法

方　法	说　明
forward(java.lang.StringrelativeUtlpath)	把网页转发到另一个页面或者 Servlet 组件上
getAttribute(java.lang.Stringname[,int scope])	scope 参数是可选的，该方法用来检索一个特定的已经命名对象的范围，并且还可以通过调用 getAttributeNameInScope()方法，检索某个特定范围的每个属性 String 字符串名称的枚举
getException()	返回当前的 exception 对象
getRequest()	返回当前的 request 对象
getResponse()	返回当前的 response 对象
getServletConfig()	返回当前页面的 servletConfig 对象
Invalidate()	返回 servletContext 对象，全部销毁
setAttribute()	设置默认网页范围或特定对象范围之中的已命名对象
removeAttribute()	删除默认网页范围或特定对象范围之中的已命名对象

pageContext 对象在实际 JSP 开发应用中很少使用，因为 request 和 response 等对象可以直接调用方法进行使用，如果通过 pageContext 对象来调用其他对象有些麻烦。

2. 应用和请求页面的 page 对象

page 对象是为了执行当前网页应答请求而设置的 Servlet 类的实体，即显示 JSP 页面本身，只有在 JSP 网页内才是合法的，类似于类中的 this 指针。page 隐含对象本质上包括当前 Servlet 接口引用的变量，因此该对象对于开发 JSP 比较有用。page 对象比较常用的方法如表 7-19 所示。

表 7-19　page 对象的常用方法

方　法	方 法 描 述
getClass()	返回当前 Object 的类
hashCode()	返回此 Object 的哈希代码
toString()	将此 Object 类转换成字符串
Equals(Object o)	比较此对象和指定的对象是否相等
copy(Object o)	把此对象赋值到指定的对象中去
Clone()	对此对象进行克隆
notify()	唤醒一个等待的线程
wait()	使一个线程处于等待直到被唤醒

例 7.19　一个使用 page 对象的例子。设定 page 指令的 info 属性为 page 对象示例，然后使用方法 getServletInfo 获取 info 属性的值。代码如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
```

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>page 对象示例</title>
</head>
<body>
<%@ page import = "java.lang.Object" %>
<%@ page import = "javax.servlet.jsp.*" %>
  <center><h3>Page 内置对象的实例</h3></center>
 <%!Object obj;%>
  getClass : <%=page.getClass()%>
  <br>
  hashCode : <%=page.hashCode()%>
  <br>
  toString : <%=page.toString()%>
  <br>
  equals : <%=page.equals(obj) %>
  <br>
  equlas2 : <%=page.equals(this)%><br/>
  <h2>page 隐含对象</h2>
 Page Information= <%=((HttpJspPage)page).getServletInfo()   %><br/>
</body>
</html>
```

运行结果如图 7-61 所示。

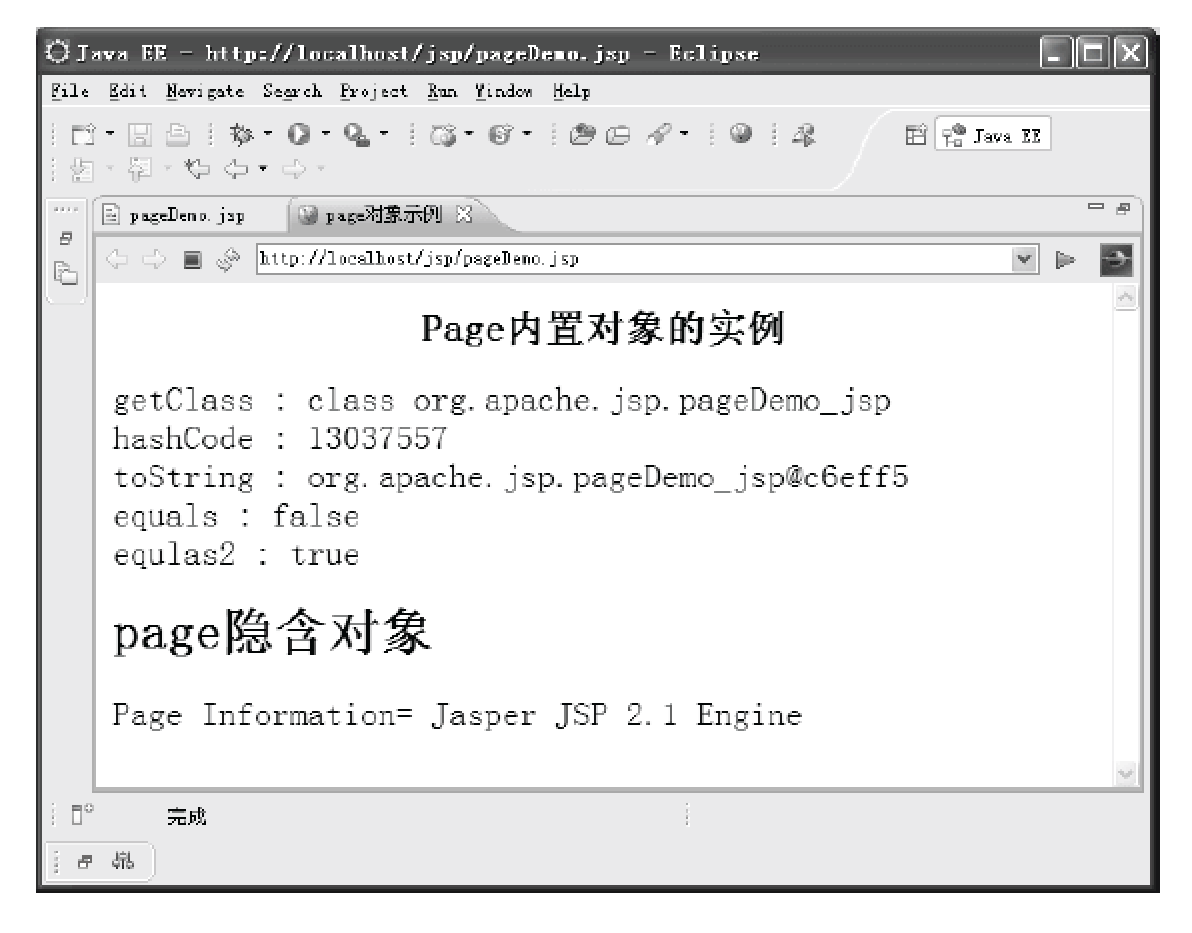


图 7-61　运行结果

3．获取异常信息的 exception 对象

exception 内置对象用来处理 JSP 文件执行时发生的所有错误和异常。exception 对象和 Java 的所有对象一样，都具有系统的继承结构，exception 对象几乎定义了所有的异常情况，它和常见的错误有所不同。这里的错误指的是可以预见的，并且知道如何解决的情况，一般在编译时可以发现。exception 对象不能在一般的 JSP 文件中直接使用，如果要使用，必须在 Page 指令中设定。设定代码为“<%@page isErrorpage="true"%>”。exception 对象比较



常用的方法如表 7-20 所示。

表 7-20　exception 对象的常用方法

方　法	说　明
getMessage()	返回异常消息字符串
getLocalizedMessage()	返回本地化语言的异常错误
printfStackTrace()	显示异常的栈跟踪轨迹
toString()	返回关于异常错误的简单信息描述
fillInStackTrace()	重写异常错误的栈执行轨迹

例 7.20　一个获取异常信息的 exception 对象示例。

通过 exception 异常对象将系统出现的异常转向到其他页面。

下面是 exception.jsp 页面，它是一个有错误的页面。设置错误的页面是 error.jsp，其代码如下：

```
<%@ page language="java" contentType="text/html; charset=gb2312"
    pageEncoding="gb2312" errorPage="error.jsp"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>exception 错误</title>
</head>
<body>
结果=
<%
        int a=100;
        int b=0;
        out.println(a/b);
%>
</body>
</html>
```

其对应的 error.jsp 的内容是：

```
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312" isErrorPage="true" import="java.io.*" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=GB2312">
<title>exception 错误提示页面</title>
</head>
<body>
```




```
<%=exception%> <br/>
错误提示为：<%=exception.getMessage()%><br/>
</body>
</html>
```

运行 exception.jsp 的结果如图 7-62 所示。

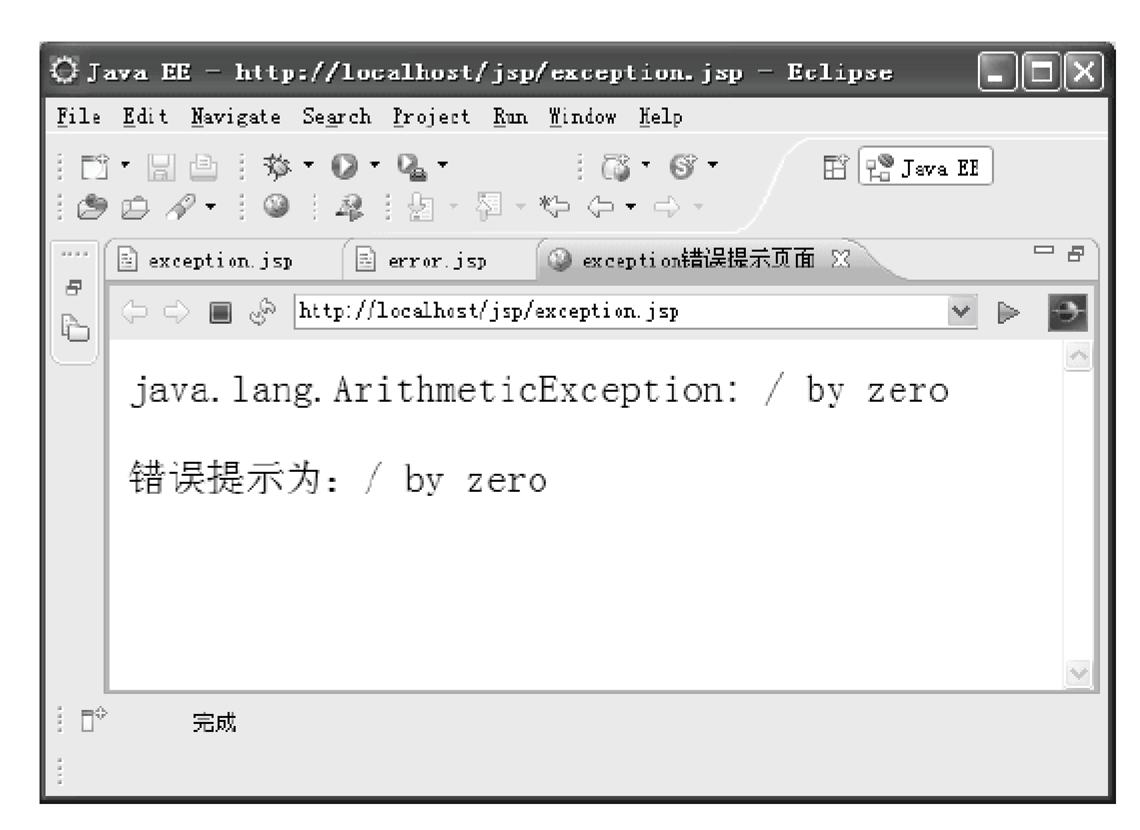


图 7-62　运行 exception.jsp 的结果

7.7　JDBC 技术

JDBC（Java DataBase Connectivity）是 Sun 公司制定的用于 Java 语言进行数据库连接的技术，它是独立于数据库管理系统的，JDBC API 提供了标准的应用程序接口。

7.7.1　JDBC 技术简介

JDBC 和 ODBC 作用非常相似，它们在应用程序和数据库之间起到桥梁的作用。JDBC 虽然和 ODBC 相似，但是仍有一些区别。ODBC 是微软公司的开放式数据库连接接口（Open DataBase Connectivity），它具有任何平台和任何数据库的更加广泛的数据库连接能力。JDBC 中 4 种驱动程序中的 JDBC-ODBCBridge 驱动就是通过 ODBC 实现所有的数据库的连接访问。

ODBC 驱动程序必须安装在客户端的机器上，而 JDBC 可以直接在本地或远程数据库服务器上执行。JDBC 安装部署简单，在安全性、易用性、维护性等方面都比 ODBC 要高。关于 ODBC 的知识，读者可参考其他书籍。在 JSP 中进行 Web 开发时访问数据库主要是通过 JDBC 来实现的。

7.7.2　JDBC 驱动程序

JDBC 驱动程序用于解决应用程序与数据库通信的问题，它可分为 JDBC-ODBC Bridge、





JDBC-Native API Bridge、JDBC-middleware 和 PureJDBC-Driver 4 种，下面分别进行介绍。

1．JDBC-ODBC Bridge

JDBC-ODBC Bridge 是通过本地的 ODBC Driver 连接到 RDBMS 上的，这种连接必须将 ODBC 二进制代码加载到使用该驱动程序的每个客户机上，所以这种类型的驱动程序最适合于企业网，或者利用 Java 编写的 3 层结构的应用程序服务器代码。

2．JDBC-Native API Bridge

JDBC-Native API Bridge 驱动通过调用本地的 native 程序实现数据库连接，这种类型的驱动程序把客户机 API 上的 JDBC 调用转换为 Oracle、Sybase、Informix、DB2 和 DBMS 的调用。需要注意的是，与 JDBC-ODBC Bridge 驱动程序一样，驱动程序要求将某些二进制代码加载到每台客户机上。

3．JDBC-middleware

JDBC-middleware 驱动是一种完全利用 Java 编写的 JDBC 驱动，这种程序将 JDBC 转换为与 OBMS 无关的网络协议，再将这种程序通过网络服务器转换为 DBMS 协议，这种网络服务器中间件能够将纯 Java 客户机连接到不同的数据库上。为了使这些产品也支持 Internet 访问，必须处理 Web 所提供的安全性、通过防火墙的访问等方面的额外要求。

4．PureJDBC-Driver

PureJDBC-Driver 驱动是一种完全利用 Java 编写的 JDBC 驱动，这种类型的驱动程序将 JDBC 调用直接转换为 DBMS 所使用的网络协议，这允许从客户机上直接调用 DBMS 服务器，是 Internet 访问的一个很实用的解决方法。

7.7.3 JDBC 中的常用接口

JDBC 提供了许多接口和类，通过它们可以实现与数据库的通信。

1．驱动程序接口 Driver

每种数据库的驱动程序都应该提供一个实例 java.sql.Driver 接口的类，在加载时，应创建自己的实例并向 java.sql.DriverManager 类注册该实例。通常通过 java.sql.Class 类的静态方法 forName(String className)加载要连接数据库的 Driver 类。成功加载后，会将 Driver 类的实例注册到 DriverManager 类中，若加载失败，将抛出 ClassNotFoundException 异常，即未找到指定的 Driver 类的异常。

2．驱动程序管理器 driverManager

java.sql.DriverManager 类负责管理 JDBC 驱动程序的基本服务，是其管理层，负责跟踪可用的驱动程序，并在数据库和驱动程序之间建立连接。此外，DriverManager 类也处理如驱动器程序登录时间限制及登录和跟踪消息的显示等工作。成功地加载 Driver 类并在

DriverManager 类中注册后，DriverManager 类即可用来建立数据库连接。表 7-21 列出了 DriverManager 类提供的常用方法。

表 7-21　DriverManager 类的常用方法

方　法	功　能
getConnection(String url,String user, String password)	为静态方法，用来获得数据库连接，有 3 个入口参数：URL、用户名和密码，返回值类型为 java.sql.Connection
setLoginTimeout(int seconds)	为静态方法，用来设置每次的等待建立数据库连接的最长时间
setLogWriter(java.io.printWriter out)	为静态方法，用来设置日志的输出对象
printIn(String message)	为静态方法，用来输出指定消息到当前的 JDBC 日志流

3. 数据库连接接口 Connection

java.sql.Connection 接口负责与特定数据库的连接，在连接的上下文中可以执行 SQL 语句并返回结果，还可以通过 getMetaData()方法获得由数据库提供的相关信息，如数据表、连接信息等信息。Connection 接口提供的常用方法如表 7-22 所示。

表 7-22　Connection 接口的常用方法

方　法	功　能
creatStatement()	创建并返回一个 Statement 实例，通常在执行无参数的 SQL 语句时创建该实例
prepareStatement()	创建并返回一个 prepareStatement 实例，通常在执行包含参数的 SQL 语句时创建该实例，并对 SQL 语句进行预编译处理
prepareCall()	创建并返回一个 prepareCall 实例，通常在调用数据库存储时创建该实例
setAutoCommit()	设置当前 Connection 实例的自动提交模式，默认为 true，即自动将更改同步到数据库中，如果设为 false，需要通过执行 Commit()和 Rollback()方法手动将更改同步到数据库中
getAutoCommit()	查看当前的 Connection 实例是否处于自动提交模式，如果是则返回 true，否则返回 false
setSavepoint()	在电脑当前事务中创建并返回一个 Savepoint 实例，但当前的 Connection 实例不能处于自动提交模式，否则将抛出异常
release Savepoint()	从当前事务中移除指定的 Savepoint 实例
setReadOnly()	设置当前的 Connection 实例的读取模式，默认为非只读模式，不能在事务中执行该操作，否则将抛出异常
isReadOnly()	查看当前的 Connection 实例是否为只读模式，如果是则返回 true，否则返回 false
isClosed()	查看当前的 Connection 实例是否被关闭，如果是则返回 true，否则返回 false
Commit()	将从上一次提交或者回滚以来进行的所有更改同步到数据库，并释放 Connection 实例当前拥有的所有数据库锁定
Rollback()	取消当前事务中的所有更改，并释放 Connection 实例当前拥有的所有数据库锁定；但只能在非自动提交模式下使用
Close()	立即释放 Connection 实例占用的数据库和 JDBC 资源，即关闭数据库连接



4. 执行 SQL 语句接口 Statement

Statement 对象主要用于执行 SQL 语句，对数据库进行操作。可利用其 creatStatement 方法创建一个 Statement 对象。java.sql.Statement 接口可用来执行静态的 SQL 语句，并获得一个记录结果集的 ResultSet。Statement 接口提供的常用方法如表 7-23 所示。

表 7-23　Statement 接口的常用方法

方　法	功　能
Cancle()	用来取消一个线程正在执行的语句
execute()	用来执行一条可能返回多个结果的 SQL 语句
close()	用来关闭 Statement 对象，释放与该句有关的 JDBC 资源，如果它有相应的结果集 ResultSet，则也会被关闭
executeBatch()	执行 batch 中的所有 SQL 语句，如果全部执行成功，则返回由更新计数组成的数组，其排列顺序与 SQL 语句的添加顺序对应
clear Batch()	清除位于 batch 中的所有 SQL 语句，如果驱动程序不支持批量处理则抛出异常
executeUpdate()	执行静态的 INSERT、UPDATE 或 DELETE 语句，并返回一个 int 型数值，为同步更新记录的条数
add Batch()	将指定的 SQL 命令添加到 batch 中，如果驱动程序不支持批量处理则抛出异常

5. 执行动态 SQL 语句接口 PreparedStatement

PreparedStatement 继承于 Statement，是 Statement 接口的扩展，用来执行动态的 SQL 语句，创建带有参数的 SQL 语句对象。通过 PreparedStatement 实例执行的动态 SQL 语句，将被预编译并保存到 PreparedStatement 实例中，可以反复并且高效地执行该 SQL 语句。PreparedStatement 的使用方法如下：

```
preparedStatement ps=connection
preparedStatement("select*from table-name where id>? And(name=? or name=?)");
ps.setInt(1,1);
ps.setString(2, "wgh");
ps.setObject(3, " sk");
Resultset rs =ps.executeQuery();
```

PreparedStatement 接口提供的常用方法如表 7-24 所示。

表 7-24　PreparedStatement 接口的常用方法

方　法	功　能
executeQuery()	执行前面包括参数的动态 SELECT 语句，并返回一个永远不能为 null 的 ResultSet 实例
executeUpdate()	执行前面包括参数的动态 INSERT、UPDATE 或 DELETE 语句，并返回一个 int 型数值，为同步更新记录的条数
clearParameters()	清除当前所有参数的值
close()	关闭 ResultSet 对象并释放资源

续表

方　法	功　能
setXxx()	为指定参数设置 Xxx 型值
execute()	用来执行一条可以返回多个结果的 SQL 语句，返回值是一个布尔值
setBoolean()	设置一个参数的布尔值

6. 执行存储过程接口 CallableStatement

Java.sql.CallableStatement 接口继承 PreparedStatement 接口，是 PreparedStatement 接口的扩展，用来执行 SQL 的存储过程。

JDBC API 定义了一套存储过程 SQL 转义语法，该语法允许对所有的 RDBMS 通过标准方式调用存储过程。该语法定义了两种形式，分别为包含结果参数和不包含结果参数。如使用结果参数，必须将其注册为 OUT 型参数，参数是根据定义位置按顺序引用的，第一个参数的索引为 1。CallableStatement 可以返回一个或多个 ResultSet 实例。处理多个 ResultSet 对象的方法是从 Statement 中继承来的。CallableStatement 接口提供的常用方法如表 7-25 所示。

表 7-25　CallableStatement 接口的常用方法

方　法	功　能
getInt()	获取一个参数的整数值
getLong()	获取一个参数的长整数值
getShort()	获取一个参数的短整数值
getString()	获取一个参数的字符串值
getDouble()	获取一个参数的双精度值
getFloat()	获取一个参数的单精度值
getByte()	获取一个参数的 Byte 值
getDate()	获取一个参数的日期值
getBoolean()	获取一个参数的 Boolean 值

7.7.4　JDBC 连接数据库的方法

应用程序访问数据库服务器，首先要进行的是连接数据库。使用 JSP 开发 Web 应用程序，常使用 JDBC 来实现数据库的连接访问。JSP 可以直接使用它们来访问数据库。下面以 SQL Server 数据库为例介绍几种常用数据库的 JDBC 连接方法。

1. SQL Server 2000 数据库

```
<%@ page contentType="text/html;charset=gb2312" import="java.sql.* "%>
<html>
<head>
<title>JDBC 连接:SQL Server 2000 数据库</title>
</head>
```



```
<body>
<%
//Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");//载入驱动程序
//String url="jdbc:odbc:bb; "//bb 为 OBDC 数据源名称
Class.forName("com.microsoft.jdbc.sqlserver.SQLServerDriver").newInstance();
String
url= ".jdbc: microsoft:sqlserver://localhost:1433;DatabaseName=data"; //连接字符串
//data 为数据库名称
String username"";//用户名
String password"";//用户密码
Connection con=DriverManager.getConnection(url, username, password);//建立数据库连接
Statement stmt=con.createStatement(ResultSet.TYPE-SCROLL-INSENSITIVE,
                                   ResultSet.CONCUR-READ-ONLY);
String sql="select*from student";
ResultSet rs=stmt.executeQuery(sql);
While(rs.next()){
System.out.println("数据库服务器连接成功。");
Rs.close();
Stmt. close();
Con. close();
%>
</body>
</html>
```

2．MySQL 数据库

```
<%@ page contentType="text/html;charset=gb2312" import="java.sql.* "%>
<html>
<head>
<title>JDBC 连接:MySQL 数据库</title>
</head>
<body>
<%
Class.forName("org.gjt.mm.mysqlDriver").newInstance();
String url= "jdbc: mysql://locallhost/StudentData?user=username&"+
"password=password&useUnicode=true;characterEncoding=8859-1; "//连接字符串
// StudentData 为数据名，username 为用户名，password 为用户密码
Connection con=DriverManager.get Connection(url);//建立数据库连接
Statement stmt=con.createStatement(ResultSet.TYPE-SCROLL-INSENSITIVE,
                                   ResultSet.CONCUR-READ-ONLY);
String sql="select*from student";
ResultSet rs=stmt.executeQuery(sql);
While(rs.next()){
System.out.println("数据库服务器连接成功。");
Rs.close();
Stmt. close();
```

```
Con. close();
%>
</body>
</html>
```

3．Oracle 数据库

```
<%@ page contentType="text/html;charset=gb2312" import="java.sql.* "%>
<html>
<head>
<title>JDBC 连接:Oracle 数据库</title>
</head>
<body>
<%
Class.forName("Orcel.jdbc.driver.orcalDriver"). newInstance();
String url=" jdbc:orcal:thin:@localhost:1521: StudentData";//连接字符串
//StudentData 为数据库名称
String username; ""//用户名
String password; ""//用户密码
Connection con=DriverManager.getConnection(url, username, password);//建立数据库连接
Statement stmt=con.createStatement(ResultSet.TYPE-SCROLL-INSENSITIVE,
                                    ResultSet.CONCUR-READ-ONLY);
String sql="select*from student";
ResultSet rs=stmt.executeQuery(sql);
While(rs.next()){
System.out.println("数据库服务器连接成功。");
Rs.close();
Stmt. close();
Con. close();
%>
</body>
</html>
```

7.8　JavaBean 封装数据库

使用 JSP 开发 Web 应用程序，常常需要访问后台数据连接，应用程序连接数据库服务器是一件频繁且消耗资源的事情，因此，常使用 JavaBean 来封装服务器数据库的连接信息，并完成数据库的访问。

JavaBean 是一种可复用的、跨平台的软件组件，它可以将所有需要实现的业务都封装在一个组件中，而 Java 程序则可以通过对 JavaBean 组件的调用，实现查询、添加、修改、删除等功能。



7.8.1　JSP 与 JavaBean 的关系

如果把全部的业务逻辑代码都放在 JSP 页面，那么 JSP 程序就和传统的 ASP 代码没有太大区别，都会在代码维护过程中遇到“一处动，处处动”这种尴尬的情况。因此，还需要把和显示逻辑无关的业务逻辑或者数据库访问逻辑等代码用 JavaBean 封装起来。

JavaBean 从表现形式上来看，是封装了诸多逻辑的 Java 代码，从架构的角度上来讲，由于不必把业务逻辑等的代码放到 JSP 中，所以采用 JSP+JavaBean 形式开发的代码可以实现良好的“逻辑分离”作用。

从功能上讲，JavaBean 是 Java 的可重用的组件技术，其中可以抽象出具有通用功能的模块和代码，使用 JavaBean 不仅能缩短开发时间，还可以提高代码的架构，为 Web 应用带来了更多的可伸缩性。

JSP、JavaBean 和数据库之间的关系如图 7-63 所示。

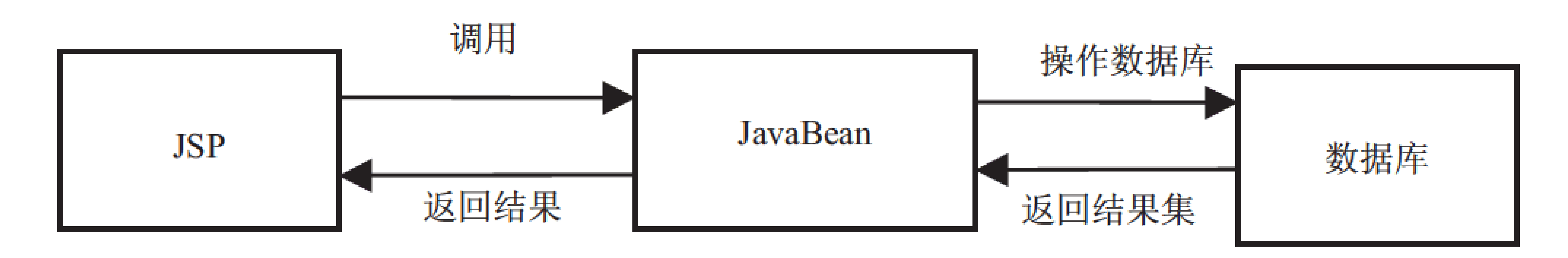


图 7-63　JSP、JavaBean 和数据库之间的关系

7.8.2　JavaBean 组件

因为 JavaBean 是一种用于构建可重用组件的 Java 类库，所以，一般把数据库访问的业务逻辑以 JavaBean 的形式封装起来，这样就可以将 JSP 页面和 JavaBean 有效地分离，加强了组件的可重用性。

首先，在 Eclipse 中创建的一个动态网站项目（Dynamic Web Project），在 JavaBeanDB 项目中的 Java Resources 下新建一个包（package），如 test；然后在其下新建一个 Java 类，如 Dbconn；然后在 DBconn.java 中输入以下代码：

```
package test;
import java.sql.*;
public class DBconn {
        static Connection conn=null;
        Statement stmt=null;
        ResultSet rs=null;
        public DBconn(){}
        public   Connection getConnect(){
              try {
                    Class.forName("net.sourceforge.jtds.jdbc.Driver");
```




```
        } catch (ClassNotFoundException e1) {
            System.out.println("Not Found The Class of The net.sourceforge.jtds.jdbc.Driver");
            e1.printStackTrace();
        }
        String url = "jdbc:jtds:sqlserver://localhost:1433/StudentInfo";
        String user="sa";
        String pwd="sa";
        try {
            conn=DriverManager.getConnection(url,user,pwd);
        } catch (SQLException e) {
            System.out.println("无法获得 Connection 对象！ ");
            e.printStackTrace();
        }
    return conn;
    }
    public ResultSet select(String sql){
    try{
    stmt=conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_
READ_ONLY);
        rs=stmt.executeQuery(sql);
    }catch(Exception e){
        e.printStackTrace();
    }
    return rs;
    }
    public int insert(String sql){
    try{
        stmt=conn.createStatement();
        stmt.execute(sql);//executeInsert
    }catch(Exception e){
        e.printStackTrace();
    }
    return 1;
    }
    public int delete(String sql){
    try{
        stmt=conn.createStatement();
        stmt.executeUpdate(sql);
    }catch(Exception e){
        e.printStackTrace();
    }
    return 1;
    }
    public int update(String sql){
    try{
        stmt=conn.createStatement();
```



```
			stmt.executeUpdate(sql);
		}catch(Exception e){
			e.printStackTrace();
		}
		return 1;
		}
		public int count(String sql){
		int num=0;
		try{
			stmt=conn.createStatement();
			rs=stmt.executeQuery(sql);
			while(rs.next()){
				num=rs.getInt(1);
			}
		}catch(Exception e){
			e.printStackTrace();
		}
		return num;
		}
		public int close(){
		try{
			if(rs!=null)
				rs.close();
			rs=null;
			if(stmt!=null)
				stmt.close();
			if(conn!=null)
				conn.close();
		}catch(Exception e ){
			e.printStackTrace();
		}finally{
			conn=null;
		}
		return 1;
		}
	}
```

注意，这里使用的是 net.sourceforge.jtds.jdbc.DriverJDBC 驱动程序的 1.2 版本，jtds 的最新版本可以从 http://sourceforge.net/projects/jtds/files/ 网站上下载。此时应确保 Java 的 CLASSPATH 中有 jtds-1.2.jar 的路径，或者将 jtds-1.2.jar 文件复制到工程的 WebContent\WEB-INF\lib 文件夹下，如图 7-64 所示。

对于数据库，选择的是 MS SQL Server 2000，用户名是 sa，密码是 sa，数据库名为 StudentInfo，其中包括一个 student 表，表中有 4 个字段：ID、Name、Age、Birthday，其结构如图 7-65 所示。

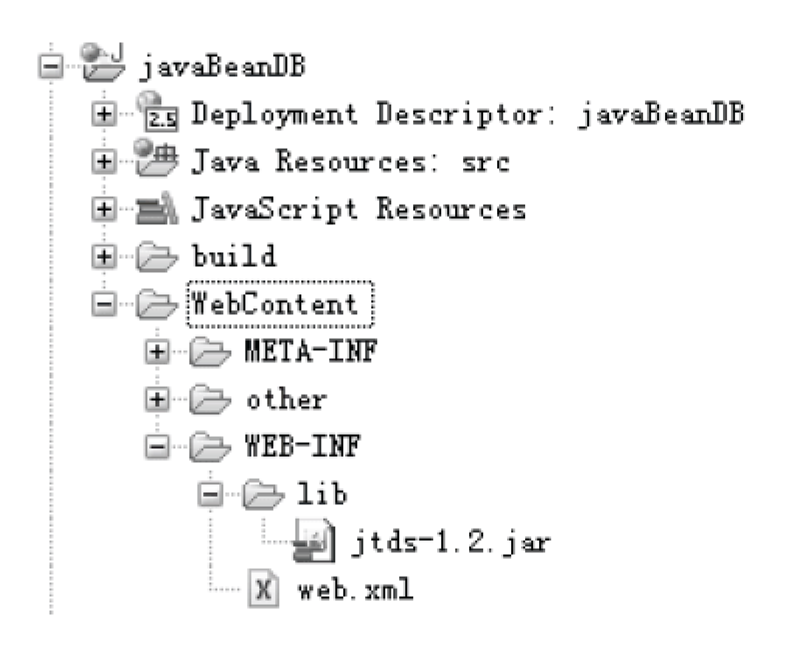


图 7-64　WebContent\WEB-INF\lib 文件夹

列名	数据类型	长度
ID	varchar	50
Name	varchar	50
Age	int	4
Birthday	datetime	8

图 7-65　student 表结构

7.8.3　页面显示

数据库交互的几个主要操作为查询、插入、更新、删除，下面给出这 4 个操作的页面代码。

（1）查询页面为 studentSelect.jsp，代码如下：

```
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=GB2312">
<title>Select Student</title>
</head>
<body>
<%@page import="java.util.*" %>
<%@page import="java.sql.*" %>
<jsp:useBean id="dbconn" class="test.DBconn" scope="page" />
<%
    dbconn.getConnect();
    String sql="select * from student";
    ResultSet rs = dbconn.select(sql);
    if (rs.wasNull()) out.println("rs is NULL..........");
%>
<table bordercolor="blue" border="1" align="center">
<tr>
<td>ID</td>
<td>Name</td>
<td>Age</td>
</tr>
<%
    while(rs.next()){
```



```
%>
<tr>
        <td><%=rs.getString("ID") %></td>
        <td><%=rs.getString("Name") %></td>
        <td><%=rs.getString("Age") %></td>
</tr>
<%
        }
%>
</table>
<%
        dbconn.close();
        out.println("Success closed!");
%>
</body>
</html>
```

运行结果如图 7-66 所示。

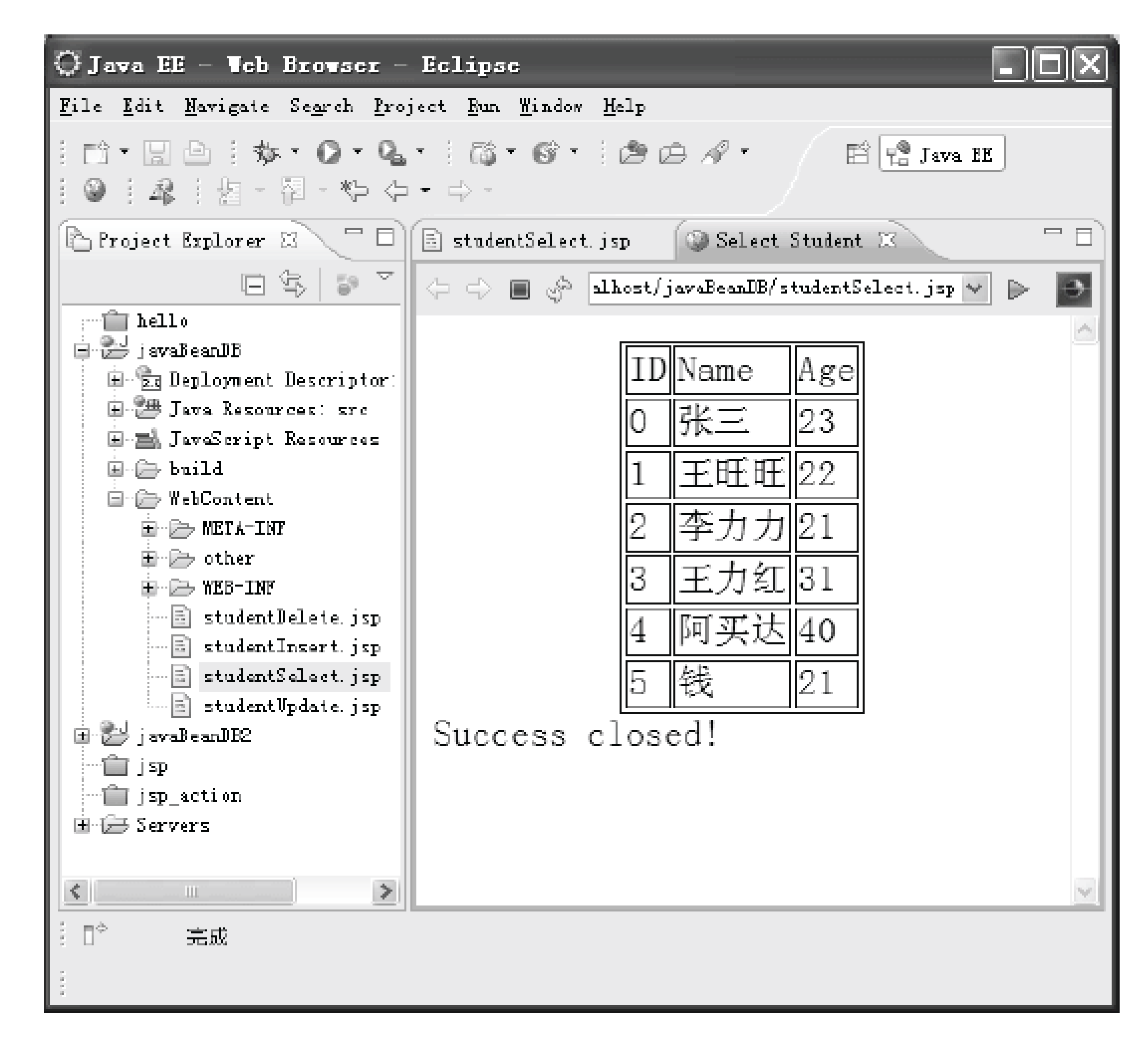


图 7-66　查询页面的运行结果

（2）插入页面为 studentInsert.jsp，代码如下：

```
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=GB2312">
<title>Insert Student</title>
</head>
<body>
```

```
<%@page import="java.util.*" %>
<%@page import="java.sql.*" %>
<jsp:useBean id="dbconn" class="test.DBconn" scope="page" />
<%
    dbconn.getConnect();
    String sql="insert into student(id,name,age,birthday) values('6','赵钱','20','1990-01-02')";
    dbconn.insert(sql);
    out.println("插入数据('6','赵钱','20','1990-01-02')成功！ ");
%>
<br/>
<a href="studentSelect.jsp">显示所有学生信息</a>
</body>
</html>
```

运行结果如图 7-67 所示。

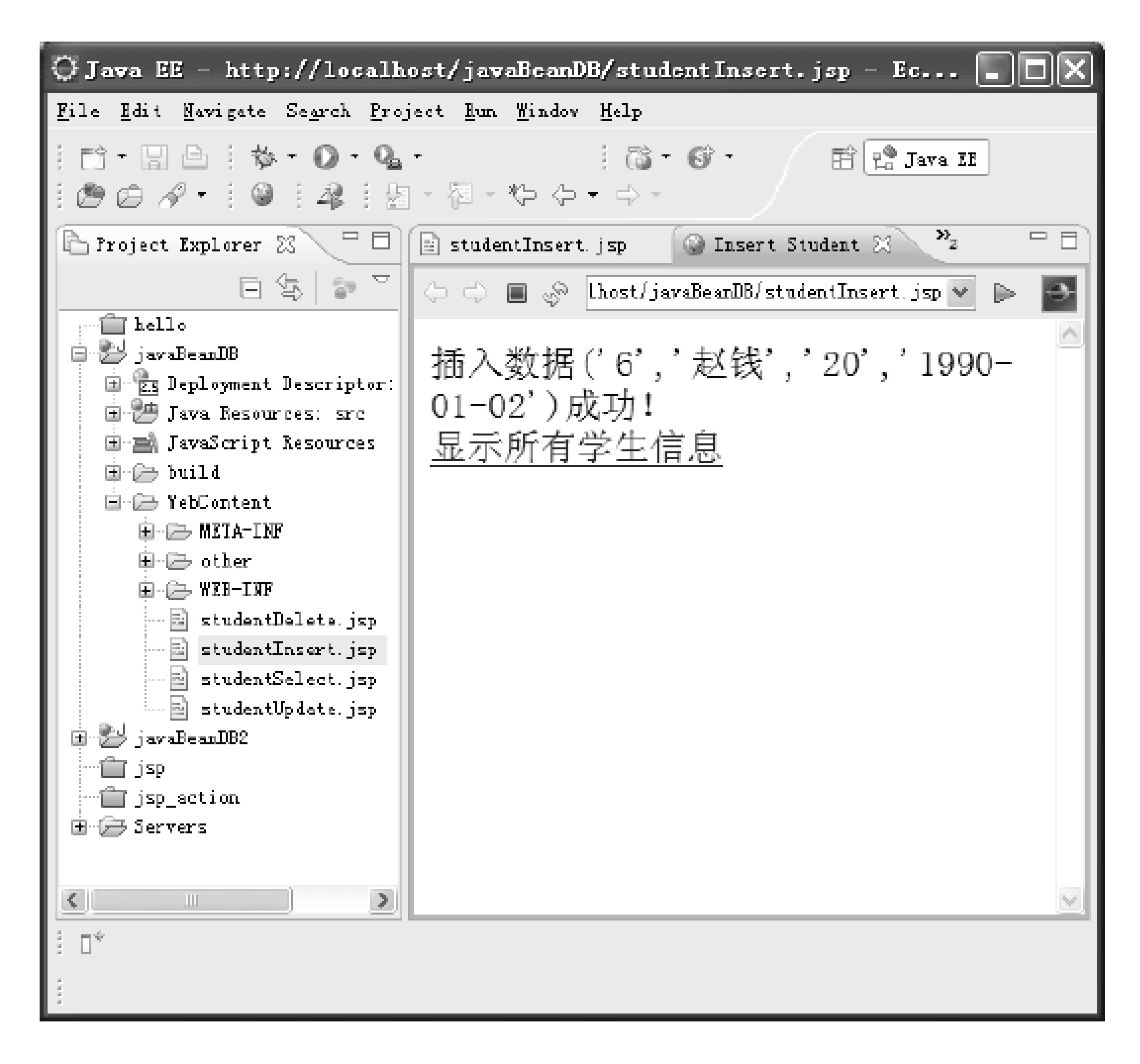


图 7-67 插入页面的运行结果

（3）更新页面为 studentUpdate.jsp，代码如下：

```
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=GB2312">
<title>Update Student</title>
</head>
<body>
<%@page import="java.util.*" %>
<%@page import="java.sql.*" %>
<jsp:useBean id="dbconn" class="test.DBconn" scope="page" />
```



```
<%
   dbconn.getConnect();
   String studentInfoSQL="select * from student";
   ResultSet rs = dbconn.select(studentInfoSQL);
   out.println("更新数据信息:<br/>");
      while(rs.next()){
            if(rs.getString("ID").equals("6")){
%>
            ID:<%=rs.getString("ID") %><br/>
            Name:<%=rs.getString("Name") %><br/>
            Age:<%=rs.getString("Age") %><br/><br/>
<p>将更新为：</p><br/>
<p>
ID:6<br/>
Name:赵倩<br/>
Age:21<br/></p>
<%
            }
      }
   String sql="update student set ID=6,Name='赵倩',Age=21 where ID=6 ";
   if(dbconn.update(sql)==1){
      out.println("更新 id=6 的数据成功！ ");
   }
   dbconn.close();%>
<br/>
<a href="studentSelect.jsp">显示所有学生信息</a>
</body>
</html>
```

运行结果如图 7-68 所示。

（4）删除页面为 studentDelete.jsp，代码如下：

```
<%@ page language="java" contentType="text/html; charset=GB2312"
    pageEncoding="GB2312"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4
/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=GB2312">
<title>Delete Student</title>
</head>
<body>
<%@page import="java.util.*" %>
<%@page import="java.sql.*" %>
<jsp:useBean id="dbconn" class="test.DBconn" scope="page" />
<%
```

```
        dbconn.getConnect();
        String studentInfoSQL="select * from student";
        ResultSet rs = dbconn.select(studentInfoSQL);
        out.println("删除数据信息:<br/>");
            while(rs.next()){
                    if(rs.getString("ID").equals("6")){
            %>
                    ID:<%=rs.getString("ID") %><br/>
                    Name:<%=rs.getString("Name") %><br/>
                    Age:<%=rs.getString("Age") %><br/>
            <%
                    }
            }
        String sql="delete from student where ID=6 ";
            if(dbconn.delete(sql)==1){
                    out.println("删除 id=6 的数据成功！ ");
            }
        dbconn.close();
    %>
    <br/>
    <a href="studentSelect.jsp">显示所有学生信息</a>
    </body>
    </html>
```

运行结果如图 7-69 所示。

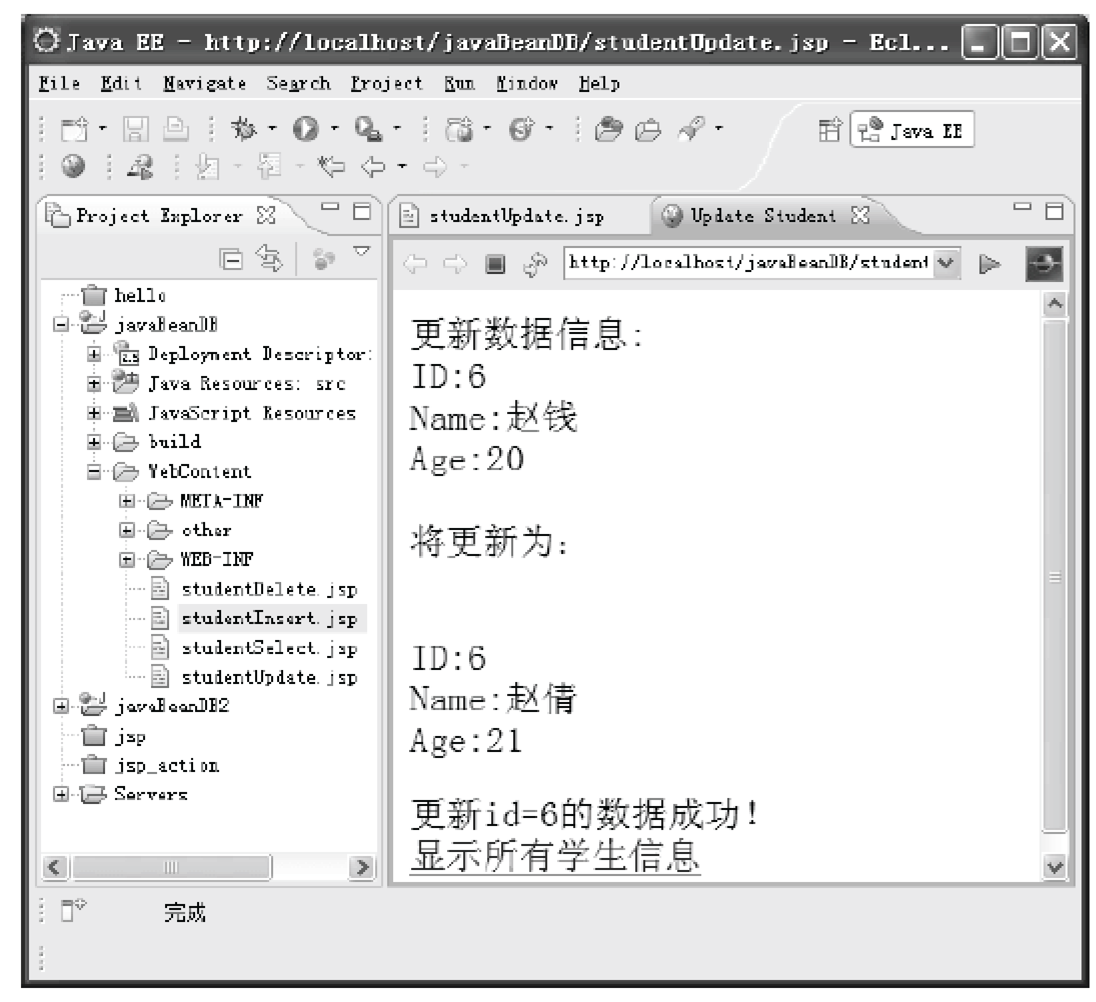


图 7-68　更新页面的运行结果

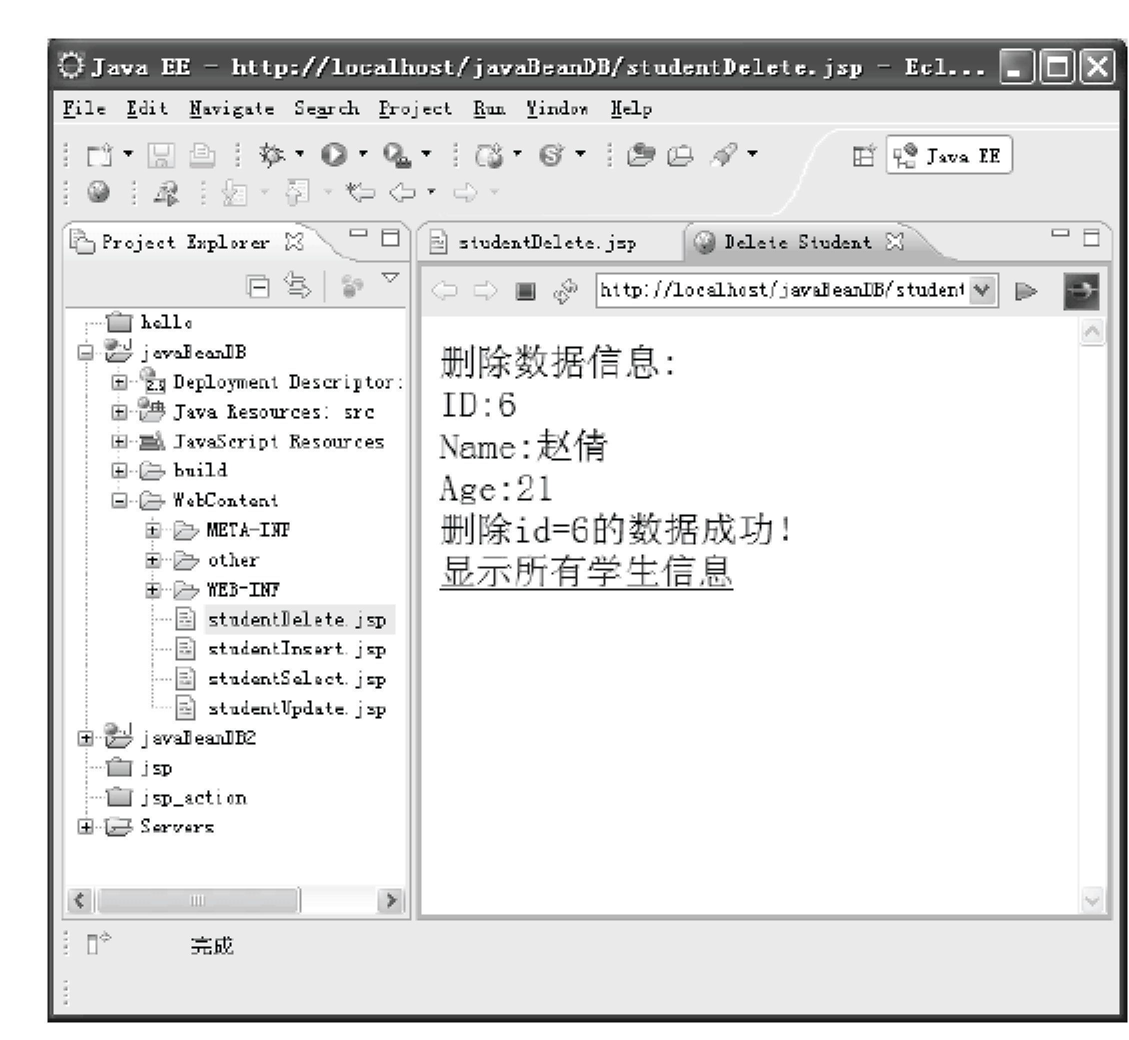


图 7-69　删除页面的运行结果

上述代码中，在插入、更新、删除页面中都需要通过查询页面来查看是否已经生效，虽然查询页面设计简单，但它是数据库查询、插入、更新、删除操作不可缺少的功能。



7.9　开 发 范 例

限于篇幅，本节以会员管理系统为例，学习如何使用 Eclipse 编写服务器端 JSP 程序。

（1）建立项目。首先使用 Eclipse 建立一个动态网站项目，命名为 MemberManager，具体步骤请参考前面章节；然后在 Eclipse 的 WebContent/WEB-INF/Lib 文件夹下加入 jtds-1.2.jar，这样就保证了之后需要进行的数据库连接部分不会出现问题。

（2）建立数据库。本例使用的是 MS SQL Server 2000，账户为 sa，密码为 sa，数据库名为 Member，其中包含一个表 Users，可以执行以下 SQL 语句来生成数据库和表：

```
CREATE DATABASE Member
USE Member
CREATE TABLE [dbo].[Users] (
        [id] [bigint] IDENTITY (1, 1) NOT NULL ,
        [username] [varchar] (20) COLLATE Chinese_PRC_CI_AS NOT NULL ,
        [password] [varchar] (20) COLLATE Chinese_PRC_CI_AS NOT NULL ,
        [rights] [char] (1) COLLATE Chinese_PRC_CI_AS NOT NULL ,
        [email] [varchar] (50) COLLATE Chinese_PRC_CI_AS NOT NULL ,
        [comments] [varchar] (50) COLLATE Chinese_PRC_CI_AS NOT NULL ,
        [loginDateTime] [datetime] NULL ,
        [loginIP] [varchar] (20) COLLATE Chinese_PRC_CI_AS NULL
) ON [PRIMARY]
GO
```

在 SQL Server 企业管理器中可以进行查看，如图 7-70 所示。

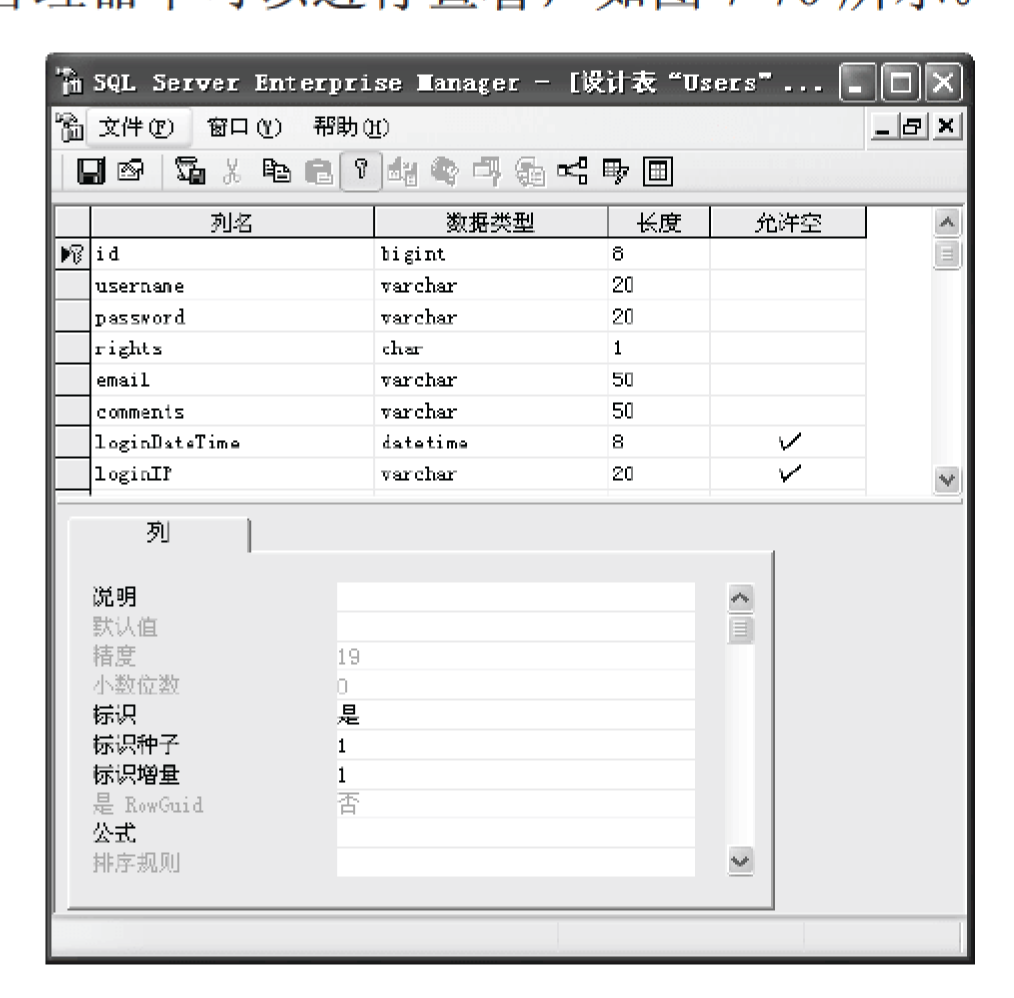


图 7-70　SQL Server 企业管理器

（3）在 Eclipse 中，在 Java Resources 中建立一个包，命名为 member，在该包下建立一个类，命名为 DBConnection.java。其代码如下：

```
package member;
import java.sql.*;
public class DBConnection {
    Connection con = null;
    Statement stmt = null;
    PreparedStatement ps = null;
    ResultSet rs = null;
    String sql = null;
    String DBDriver =  "net.sourceforge.jtds.jdbc.Driver";
    String url =  "jdbc:jtds:sqlserver://localhost:1433/Member";
    String username = "sa"; //用户名
    String password = "sa"; //用户密码
    public DBConnection() {
    }
    public void getConnection() {
          try {
          Class.forName(DBDriver); //载入驱动程序
       } catch (ClassNotFoundException e1) {
            System.out.printf("Not Found The Class of The %s\n",DBDriver);
            e1.printStackTrace();
       }
    try {
        con=DriverManager.getConnection(url,username,password);
        }catch(Exception e) {
            System.out.println("连接数据库服务器失败。");
            e.printStackTrace();
        }
    }
    public ResultSet executeQuery(String sql) {//查询数据
        try {stmt=con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
                                                ResultSet.CONCUR_READ_ONLY);
            rs=stmt.executeQuery(sql);
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        return rs;
    }
    public void closeConn() {//关闭数据库连接并释放资源
        try {
            if (rs!=null) {
                rs.close();

                rs=null;
            }
            if (stmt!=null) {
```



```
                stmt.close();
                stmt=null;
            }
            if (ps!=null) {
                ps.close();
                ps=null;
            }
            if (con!=null) {
                con.close();
                con=null;
            }
        }
        catch(Exception e) {
            e.printStackTrace();
        }
        finally {
            con=null;
        }
    }
}
```

（4）在 Eclipse 中，在 Java Resources 中的 member 包下建立另一个类，命名为 User.java。其代码如下：

```
package member;
import java.sql.*;
public class User {
    private String username; //用户名称
    private String loginIP; //用户 IP
    private String loginDateTime; //用户登录日期时间
    Connection con = null;
    Statement stmt = null;
    PreparedStatement ps = null;
    ResultSet rs = null;
    public User() {
        username = "";
        loginIP = "";
        loginDateTime = "";
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String user) {
        username = user;
    }
    public String getLoginIP() {
```

```
        return loginIP;
    }
    public void setLoginIP(String userIP) {
        loginIP = userIP;
    }
    public String getLoginDateTime() {
        return loginDateTime;
    }
    public void setLoginDateTime(String currentDateTime) {
        loginDateTime = currentDateTime;
    }
    //检查用户是否存在
    public boolean userExist(String username) {
       boolean exist = false;
       String sql = "select * from Users where username=?"; //构造查询 SQL 语句
       DBConnection dbConnection = new DBConnection();
       dbConnection.getConnection();
       try {
            con = dbConnection.con;
            ps = con.prepareStatement(sql);
            ps.setString(1, username);
            rs = ps.executeQuery();
            if (!rs.next()) {
                exist = true;
          }
        }
        catch (SQLException e) {
            e.printStackTrace();
        }
        finally {
            dbConnection.closeConn();
        }
        return exist;
     }
     //判断用户是否合法
     public boolean isValidUser(String username, String password) {
         boolean isValid = false;
         String sql = "select * from Users where username=? and password=?"; //构造查询 SQL 语句
         DBConnection dbConnection = new DBConnection();
         dbConnection.getConnection();
         try {
             con = dbConnection.con;

             ps = con.prepareStatement(sql);
             ps.setString(1, username);
             ps.setString(2, password);
```



```
            rs = ps.executeQuery();
            if (rs.next()) {
                isValid = true;
            }
        }
        catch (SQLException e) {
            e.printStackTrace();
        }
        finally {
            dbConnection.closeConn();
        }
        return isValid;
    }
    //获取用户权限
    public String userRights(String username, String password) {
        String rights = null;
        String sql = "select rights from Users where username=? and password=?";
                                                                    //构造查询 SQL 语句
        DBConnection dbConnection = new DBConnection();
        dbConnection.getConnection();
        try {
            con = dbConnection.con;
            ps = con.prepareStatement(sql);
            ps.setString(1, username);
            ps.setString(2, password);
            rs = ps.executeQuery();
            if (rs.next()) {
                rights = rs.getString("rights");
            }
        }
        catch (SQLException e) {
            e.printStackTrace();
        }finally {
            dbConnection.closeConn();
        }
        return rights;
    }
    //保存用户登录信息
    public void saveUserLoginInfo(String currentDateTime, String userIP, String username) {
        String sql = "update Users set loginDateTime = ? ,loginIP= ? where username = ?";
                                                                    //构造更新 SQL 语句
        DBConnection dbConnection = new DBConnection();
        dbConnection.getConnection();
        try {
            con = dbConnection.con;
            ps = con.prepareStatement(sql);
```

```
            ps.setString(1, currentDateTime);
            ps.setString(2, userIP);
            ps.setString(3, username);
            ps.executeUpdate();
        }
        catch (SQLException e) {
            e.printStackTrace();
        }
        finally {
            dbConnection.closeConn();
        }
    }
    //用户注册
    public boolean registerUser(String username, String password, String email) {
        String sql = "insert into Users(username,password,email) values( ? , ? , ? )";
                                                                    //构造更新 SQL 语句
        DBConnection dbConnection = new DBConnection();
        dbConnection.getConnection();
        try {
            con = dbConnection.con;
            ps = con.prepareStatement(sql);
            ps.setString(1, username);
            ps.setString(2, password);
            ps.setString(3, email);
            ps.executeUpdate();
        }
        catch (SQLException e) {
            e.printStackTrace();
            return false;
        }
        finally {
            dbConnection.closeConn();
        }
        return true;
    }

    //删除用户
    public boolean delUser(String username) {
        String sql = "delete from Users where username= ?"; //构造更新 SQL 语句
        DBConnection dbConnection = new DBConnection();
        dbConnection.getConnection();
        try {
            con = dbConnection.con;
            ps = con.prepareStatement(sql);
            ps.setString(1, username);
            ps.executeUpdate();
```



```
            }
            catch (SQLException e) {
                e.printStackTrace();
                return false;
            }
            finally {
                dbConnection.closeConn();
            }
            return true;
        }
        //获取当前日期时间
        public String getCurrentDateTime() {
            java.util.Date DateTime=new java.util.Date();//设置日期时间格式
            java.text.SimpleDateFormat currentDateTime=new java.text.SimpleDateFormat("yyyy-MM-
dd hh:mm:ss");
            return currentDateTime.format(DateTime);//获取当前日期时间
        }
    }
```

（5）在 Eclipse 中，在 WebContent 文件夹下建立另一个 jsp 页面文件，命名为 main.jsp 作为主页。其代码如下：

```
<%@ page contentType="text/html;charset=gb2312" import="java.sql.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>
<html>
<head>
<title>Home  主页</title>
</head>
<body >
<center><font size=2>
<a href="main.jsp">Home</a>  
<%if(user.userExist((String)session.getAttribute("username")))
      {
%>
      <a href="register.jsp">注册</a>
      <a href="login.jsp">登录</a>
<%
      out.println("<br/>请登录！非注册会员不可以查看信息<br/>");
      }else{
%>
      <a href="logout.jsp">注销</a>
      <br>登录用户信息一览表<br>
<%
  //获取登录用户信息
  String username=(String)session.getAttribute("username");//获取用户名
```

```
    String rights=(String)session.getAttribute("rights");//获取用户权限
    String loginDateTime=user.getLoginDateTime();//获取用户登录日期时间
    String loginIP=user.getLoginIP();//获取用户 IP
  %>
  </font><p/>

  <font size=2 color="blue">用户登录信息：用户名：<%=username%>  用户权限：
<%=rights%>  登录日期时间：<%=loginDateTime%>  用户 IP：<%=loginIP%>
</font>
  <table width="650" border="0" bgcolor="#FFFFFF" cellspacing="1" align="center">
    <tr style='font-size:9pt;color:#FF6600' bgcolor='#88D85A' align='center'>
     <td>用户名</td>
     <td>用户权限</td>
     <td>email</td>
     <td>上次登录时间</td>
     <td>上次登录 IP</td>
      <% if(rights.equals("2")){
             out.println("<td>操作</td>");
        }
        %>
    </tr>
  <%
    member.DBConnection dbConnection=new member.DBConnection();
                                          //获取数据库访问对象 DBConnection 实例 dbConnection
    dbConnection.getConnection();//调用 DBConnection 数据库连接方法
    String sql="select username,rights,email,loginDateTime,loginIP from Users";
    ResultSet rs=dbConnection.executeQuery(sql);//调用 DBConnection 数据查询方法
    //if (rs.wasNull()) out.println("rs is NULL...........");
    boolean ifNext=false;
    while (ifNext=rs.next()){
  %>
    <tr style='font-size:9pt;' align="left">
     <td><%=rs.getString("username")%></td>
     <td><%
     if(rs.getString("rights").equals("1"))out.println("普通会员");
     else if(rs.getString("rights").equals("2"))out.println("管理员");
     else out.println("权限错误");
     %></td>
     <td><%=rs.getString("email")%></td>
     <td><%=rs.getString("loginDateTime")%></td>
     <td><%=rs.getString("loginIP")%></td>
     <% if(rights.equals("2")){

      String delUserUrl=String.format("<td><a href='delUser.jsp?username=%s'>",rs.getString("username"));
      out.println(delUserUrl+"删除此用户</a></td>");
     }
```



```
 %>
  </tr>
<%}
  dbConnection.closeConn();//调用 DBConnection 关闭数据库连接方法
      }
%>
</table>
</center>
</body>
</html>
```

其页面显示为当普通用户登录时出现的显示页面，如图 7-71 所示。

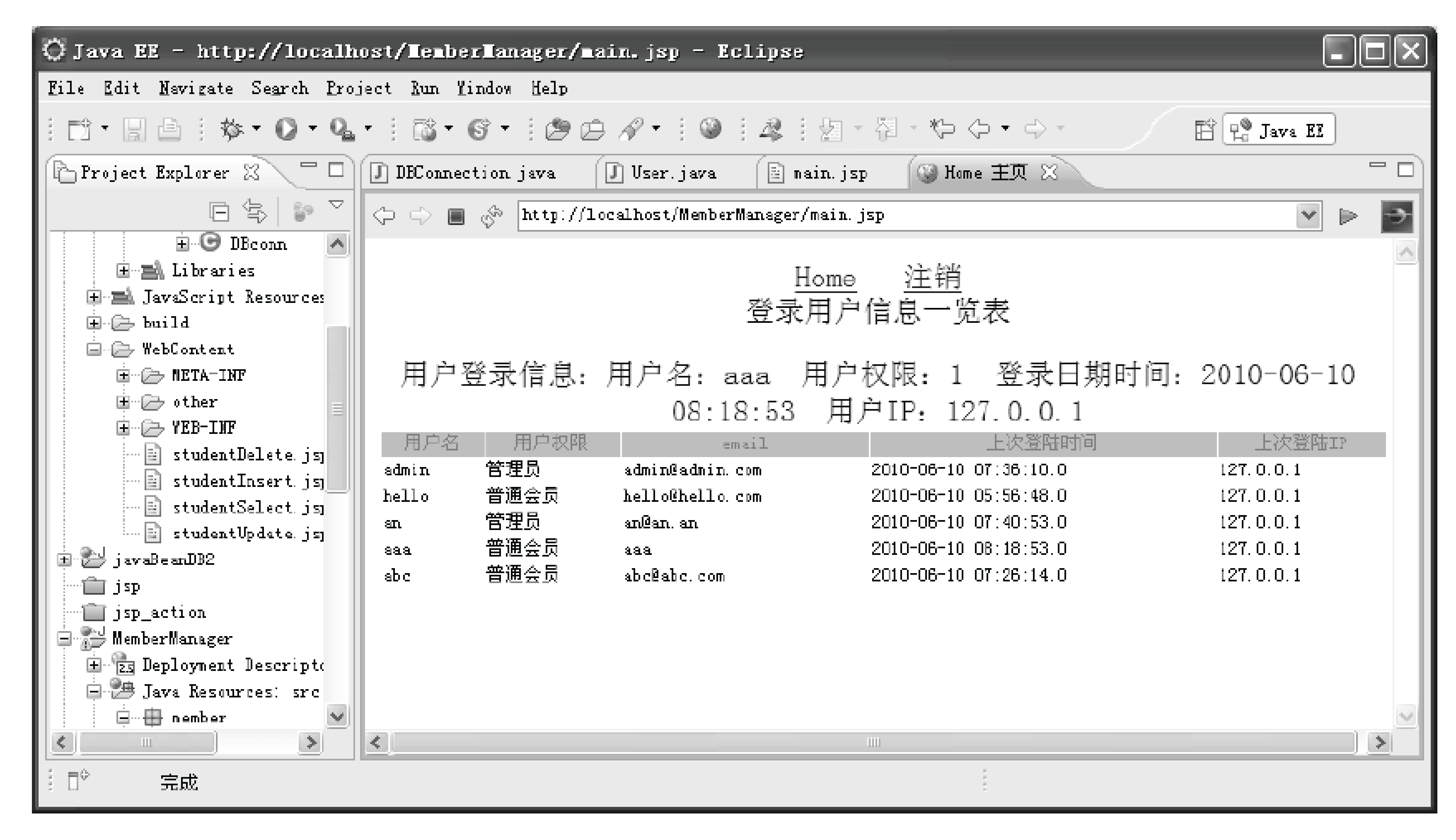


图 7-71　登录显示页面

如果是管理员，则还会在每个用户的后面多显示一个“删除此用户”链接。如果没有登录就访问了此页面，则会提示“请登录！非注册会员不可以查看信息”。

（6）新建一个 login.jsp 页面，起到会员登录的作用。其代码如下：

```
<%@ page contentType="text/html;charset=gb2312" import="java.sql.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>
<%@ page errorPage="error.jsp"%>
<html>
<head>
<title>用户登录</title>
<script language="javascript">
function checkform()//验证输入数据的合法性
{
    if (form1.username.value=="") {
        alert("用户名不能为空。");
        form1.username.focus();
        return false;
```

```
            }
            if (form1.username.value.length<1||form1.username.value.length>20) {
                alert("用户名超出了范围（1～20）。");
                form1.username.focus();
                return false;
            }
        if (form1.password.value=="") {
            alert("密码不能为空。");
            form1.password.focus();
            return false;
            }
            if (form1.password.value.length<1||form1.password.value.length>20) {
                alert("密码超出了范围（1～20）。");
                form1.password.focus();
                return false;
            }
        }
        </script>
        </head>
        <body>
        <p><b><font size=2 color='#FF6600'>用户登录</font></b></p>
        <form name="form1" method="post" action="isLogin.jsp" onsubmit="return checkform()">
          <table style='font-size:9pt;'width="320" border="0" bgcolor="#B8D85A" cellspacing="1" cellpadding=
"1" align="center">
            <tr align="center">
              <td colspan="2">
                <p><br></p>
              </td>
            </tr>
            <tr>
              <td width="140" align="right">用户名：</td>
              <td width="250" valign="top">
                <input type="text" name="username" size="16" maxlength="25">
              </td>
            </tr>
            <tr>
              <td width="140" align="right">密  码：</td>
              <td width="250" valign="top">
                <p><input type="password" name="password" size="16" maxlength="25"></p>
              </td>
            </tr>
            <tr>
              <td width="140" align="right"> </td>
              <td width="250" valign="top">
                <p/><br>
```



```
            <input type="submit" name="submit" value=" 登 录 "style='font-size:9pt;' onclick="javascript:
return(checkform());">   
            <input type="reset" name="reset" value="取 消" style='font-size:9pt;'>
          </td>
        </tr>
        <tr>
          <td colspan="2" align="center">
            <p><br>如果您还不是本站会员用户，请在此<a href="login.jsp">注册</a></p>
            <p></p>
          </td>
        </tr>
      </table>
    </form>
    </body>
    </html>
```

其运行结果如图 7-72 所示。

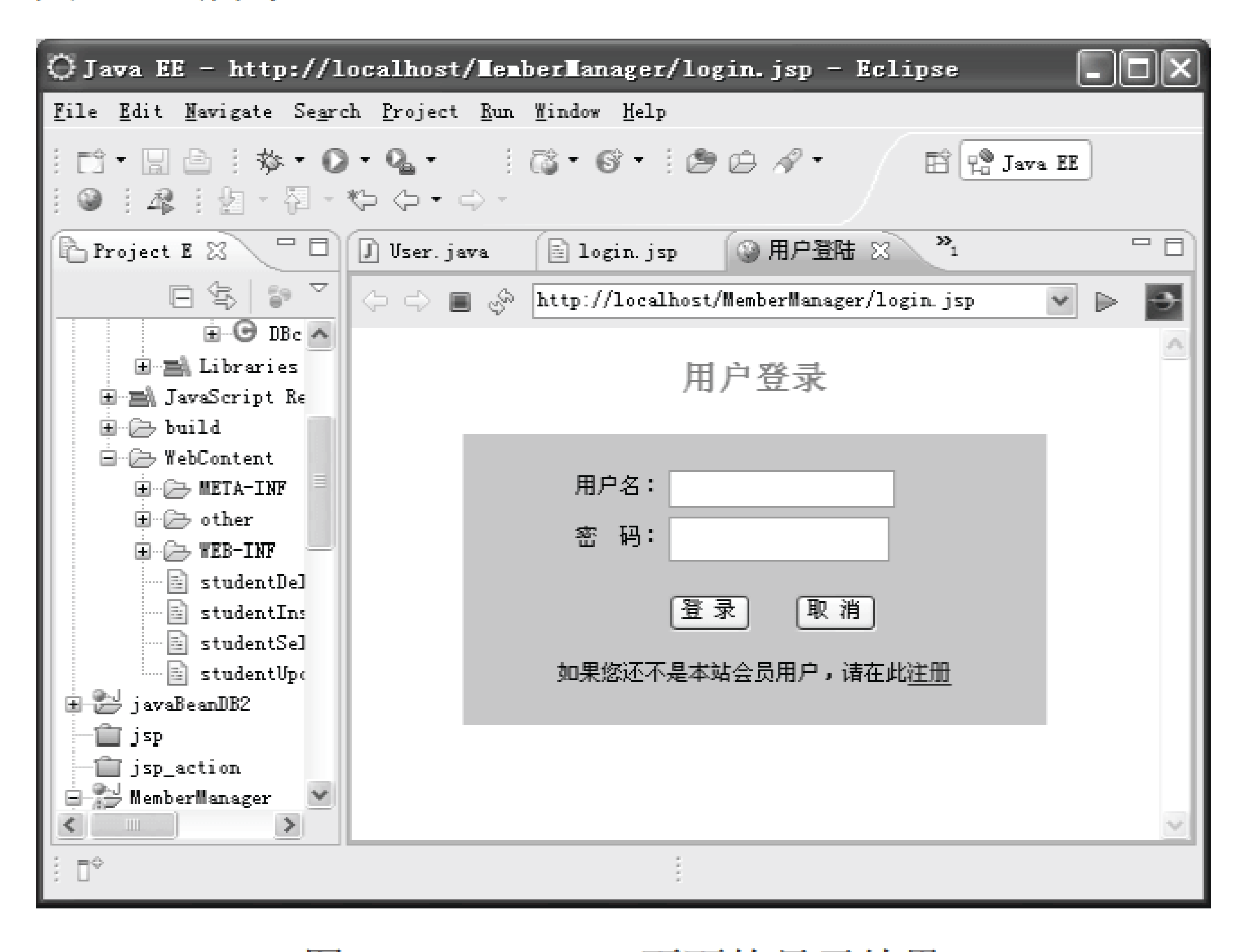


图 7-72　login.jsp 页面的显示结果

单击“登录”按钮就会跳转到 isLogin.jsp 页面进行验证，如果成功，则跳转到主页；如果失败，则跳转到显示错误的页面。isLogin.jsp 页面的代码如下：

```
<%@page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<%@ page import="java.text.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>
<%
String username,password;
  //用户登录验证
  if(request.getParameter("username")=="")  {username="NULL";password="NULL";}
```

```
else{
  username=new String(request.getParameter("username").trim().getBytes("ISO-8859-1"));
  password=new String(request.getParameter("password").trim().getBytes("ISO-8859-1"));
}
if(user.userExist(username)){
   response.sendRedirect("error.jsp?errorMessage=User is not valid.");
   }
else if (!user.isValidUser(username,password)){
     response.sendRedirect("error.jsp?errorMessage=Password is not valid.");
     }
else{
     session.setAttribute("username",username);//保存用户名
     String rights=user.userRights(username,password);//获取用户权限
     session.setAttribute("rights",rights);//设置用户权限
     String loginIP=request.getRemoteAddr();//获取用户 IP
     user.setLoginIP(loginIP);//设置用户 IP
     String loginDateTime=user.getCurrentDateTime();//获取当前日期时间
     user.setLoginDateTime(loginDateTime);//设置用户登录日期时间
     user.saveUserLoginInfo(loginDateTime,loginIP,username);//保存用户登录信息
     response.sendRedirect("main.jsp");//打开主页面
}
%>
```

（7）error.jsp 页面根据传入的参数显示错误信息，其代码如下：

```
<%@ page contentType="text/html; charset=gb2312" language="java" import="java.sql.*"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=gb2312">
<title>错误信息</title>
</head>
<body>
<%
  String error=new String(request.getParameter("errorMessage").trim().getBytes("ISO-8859-1"));
%>
<p> </p>
<center>
<font size=2>系统提示：用户名不存在或密码错误。<br>
<br><%=error%><br>
<br>如果您还不是本站会员用户，请在此<a href="register.jsp">注册</a>  <a href=
"login.jsp">重新登录</a></font>
</center>
</body>
</html>
```

error.jsp 页面的运行结果如图 7-73 所示。



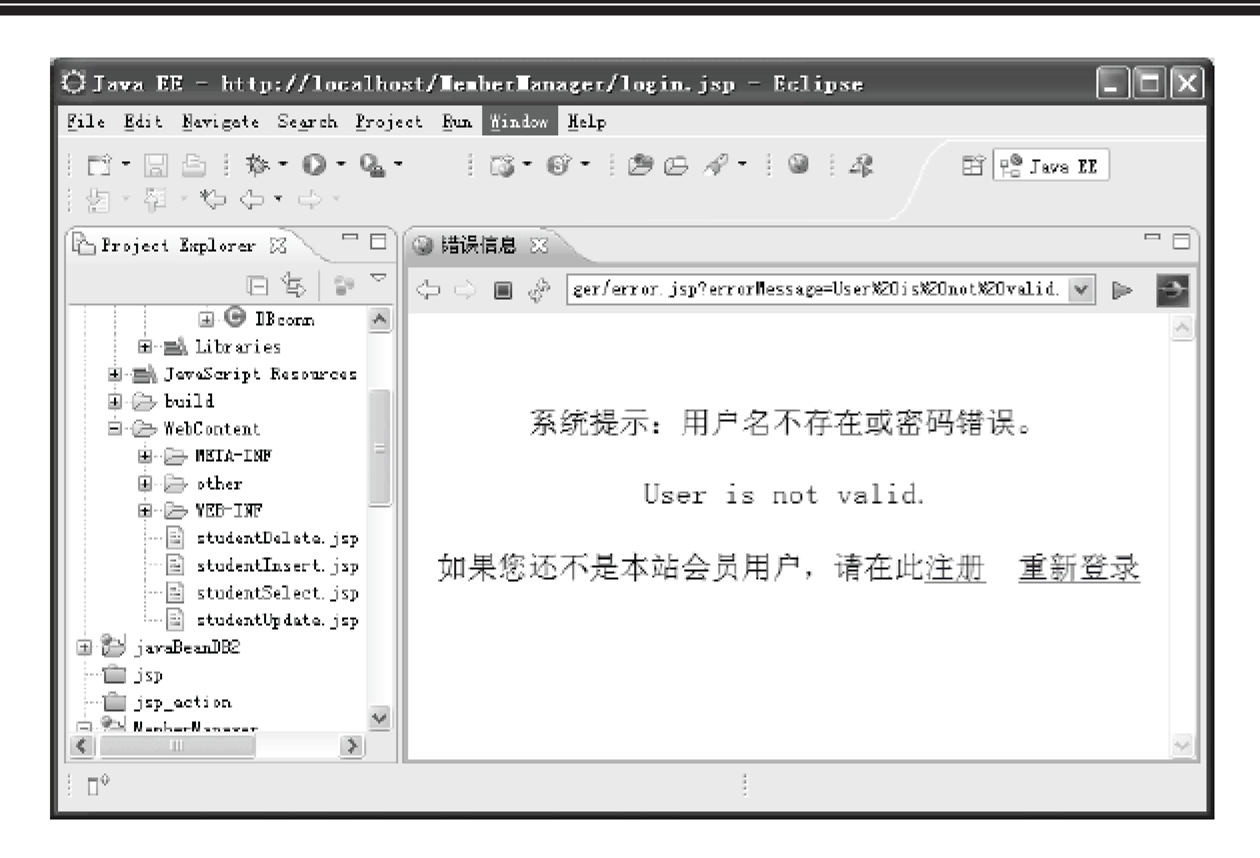


图 7-73　error.jsp 页面的运行结果

（8）register.jsp 是用户注册的页面，其代码如下：

```
<%@ page contentType="text/html;charset=gb2312" import="java.sql.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>
<%@ page errorPage="error.jsp"%>
<html>
<head>
<title>用户注册</title>
<script language="javascript">
function checkform()//验证输入数据的合法性
{
    if (form1.username.value=="") {
        alert("用户名不能为空。");
        form1.username.focus();
        return false;
    }
    if (form1.username.value.length<1||form1.username.value.length>20) {
        alert("用户名超出了范围（1～20）。");
        form1.username.focus();
        return false;
    }
   if (form1.password.value=="") {
       alert("密码不能为空。");
       form1.password.focus();
       return false;
    }
    if (form1.password.value.length<1||form1.password.value.length>20) {
        alert("密码超出了范围（1～20）。");
        form1.password.focus();
        return false;
    }
}
</script>
<style type="text/css">
```




```
.fontCenter {
        text-align: center;
        font-size: 12px;
}
.fontSmall {
        font-size: 12px;
}
</style>
</head>
<body>
<p class="fontCenter"><b><font color='#FF6600' size=2 class="fontCenter">用户注册</font></b></p>
<form name="form1" method="post" class="fontCenter" action="regok.jsp" onSubmit="return checkform()">
 <table width="300">
        <tr>
              <td width="63" class="fontSmall" >用户名</td>
          <td width="225"><input name="username" type="text"></td>
        </tr>
        <tr>
              <td class="fontSmall">密码</td>
              <td><input name="password" type="password" ></td>
        </tr>
        <tr>
              <td class="fontSmall">邮件</td>
              <td><input name="email" type="text"></td>
        </tr>
        <tr>
              <td class="fontSmall">提交</td>
              <td ><input name="register" type="submit" value="注册"></td>
        </tr>
        </table>
</form>
</body>
</html>
```

运行结果如图 7-74 所示。

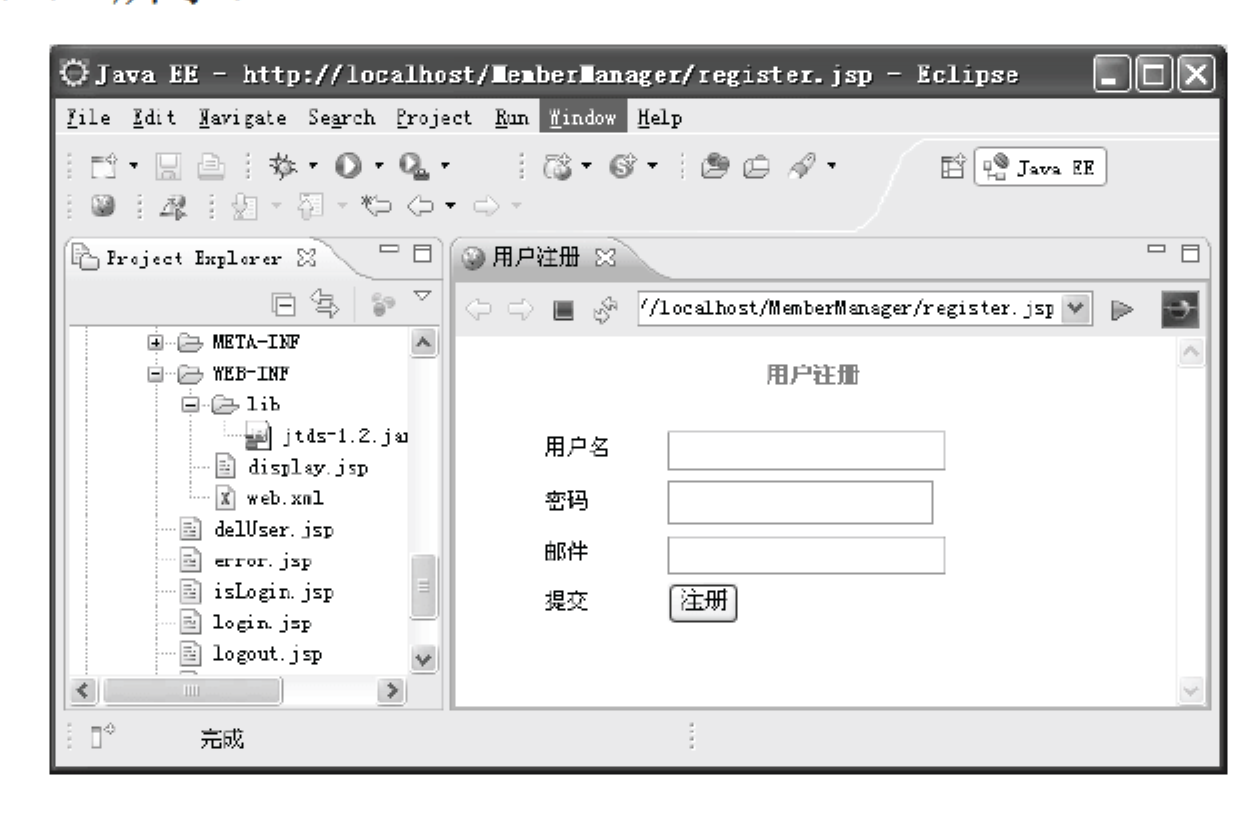


图 7-74 用户注册页面的显示结果



（9）当单击“注册”按钮后，会跳转到 regok.jsp 页面，如果注册成功，则再跳转到主页面，否则跳转到显示错误的页面。regok.jsp 页面的代码如下：

```
<%@page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<%@ page import="java.text.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>
<%
    String username,password,email;
    if(request.getParameter("username")=="")  {
            out.println("Not receive username param..");
        username="NULL";
        password="NULL";
        email="NULL";
  // response.sendRedirect("main.jsp");
 }else{
 username=new String(request.getParameter("username").trim().getBytes("ISO-8859-1"));
 password=new String(request.getParameter("password").trim().getBytes("ISO-8859-1"));
 email=new String(request.getParameter("email").trim().getBytes("ISO-8859-1"));
 }

 if(!user.userExist(username)){
 response.sendRedirect("error.jsp?errorMessage=User is exist.");
 }   else{
        if(user.registerUser(username,password,email))
            out.println("注册成功！ <br/>");
        else out.println("注册失败！ <br/>");
        String str=String.format("isLogin.jsp?username=%s&password=%s",username,password);
        response.sendRedirect(str);
  }
%>
```

（10）logout.jsp 页面可以让登录用户注销登录，其代码如下：

```
<%@page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<%@ page import="java.text.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>

<html>
<head>
<title>注销登录</title>
</head>
<body>
<center>
<%
```

```
        String str=new String();
        str=(String)session.getAttribute("username");
        if (str!=null)
        {
            if(!str.isEmpty())
            {
             session.removeAttribute("username");//清除用户名
             session.removeAttribute("rights");//清除用户权限
             out.println("注销成功<br/>");
            }else{
                    out.println("你还没有登录<br/>");
            }
        }else{
        out.println("你还没有登录<br/>");
        }
    %>
    <a href="login.jsp">登录</a><br/>
    <a href="main.jsp">返回主页</a>
    </center>
    </body>
```

error 页面的显示结果如图 7-75 所示。

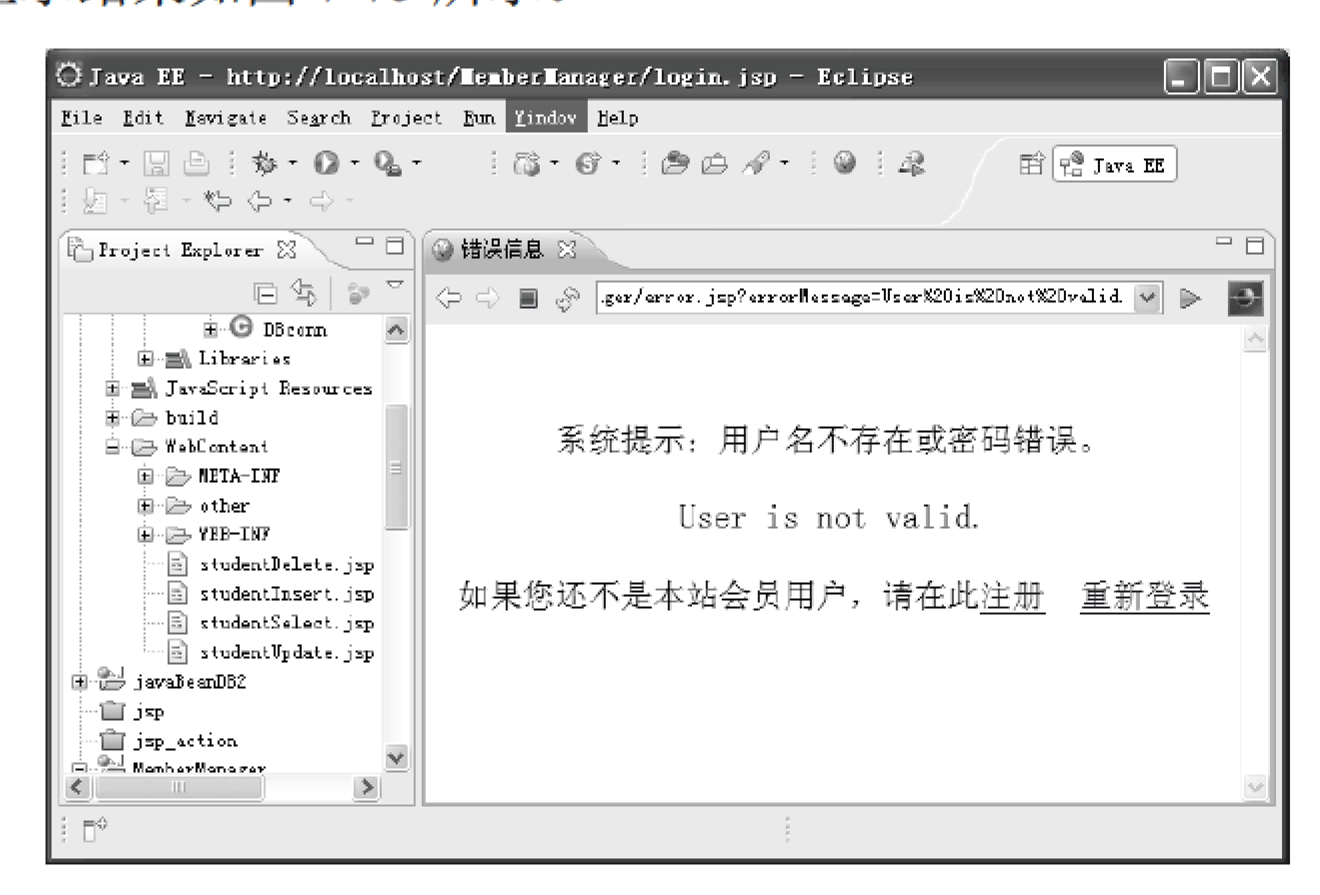


图 7-75　error 页面的显示结果

（11）delUser.jsp 根据传入的参数删除响应的用户信息，其代码如下：

```
<%@page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<%@ page import="java.text.*"%>
<jsp:useBean id="user" scope="session" class="member.User"/>
<html>
<head>
<title>管理员删除用户操作</title>
```



```
</head>
<body>
<%
String username,password;
  //用户登录验证
  if(request.getParameter("username")=="")   {username="NULL";password="NULL";
  response.sendRedirect("error.jsp?errorMessage=非法的操作.");}
  else{
   username=new String(request.getParameter("username").trim().getBytes("ISO-8859-1"));
  }
  String rights=(String)session.getAttribute("rights");
 if(rights.equals("2")){
   if(!user.userExist(username)){
              if(user.delUser(username))
                   out.println("删除用户"+username+"成功！<br/>");
              else out.println("删除用户"+username+"不成功！<br/>");
     }else{
      response.sendRedirect("error.jsp?errorMessage=User is not valid.");
   }
 }
%>
<a href="main.jsp">返回主页</a>
</body>
```

7.10　小　　结

本章主要讲解了 JSP 的概况、JDK 的获取和安装、Tomcat 服务器的下载和配置、JSP 的开发环境 Eclipse 的配置、JSP 的基础知识、JSP 的内置对象、4 种 JDBC 驱动程序（分别为 JDBC-ODBC Bridge、JDBC-Native API Bridge、JDBC-middleware 和 PureJDBC-Driver）、DriverManager 类方法、使用 JDBC 连接 4 种数据库的方法、用 JavaBean 封装服务器数据库等内容，最后用一个会员管理系统作为范例，详细介绍了如何在 Eclipse 环境下编写 JSP 动态网页。

由于篇幅有限，本章不能涵盖所有的 JSP 的内容，但对于一个初学者来说，学习本章后，会掌握 JSP 基础知识并能够编写基于 JSP 的服务器端程序。

7.11　思　考　题

1．与传统的 CGI 方式相比，JSP 具有哪些特点？

2．简述运行 JSP 程序的环境。为什么要配置环境变量？

3．Tomcat 与 Eclipse 是如何关联的？

4．JSP 包括哪些常用的指令？

5．scope 作用域的取值及有效范围是什么？

6．JDBC 的 4 种驱动程序的含义是什么？

7．如何通过 JDBC 连接 SQL Server 2000 数据库 st 中的 student 表？

8．为什么要通过 JavaBean 组件封装数据库访问？

9．JavaBean 组件与 JSP 网页之间是如何关联的？

10．设计一个基于 JSP 的学生成绩管理系统，使得管理员用户具有添加、查询、删除学生成绩权限；一般学生用户只有查询该学生成绩的权限。

参 考 文 献

1．郝兴伟．Web 技术导论．北京：清华大学出版社，2009
2．曾海．JavaScript 程序设计基础教程．北京：人民邮电出版社，2009
3．孙卫琴．Java 面向对象编程．北京：电子工业出版社，2004
4．孙卫琴．Tomcat 与 Java Web 开发及技术详解．北京：电子工业出版社，2004
5．袁建洲．JavaScript 编程宝典．北京：电子工业出版社，2006
6．曹衍龙．Ajax 编程技术与实例．北京：人民邮电出版社，2007